Sparse Linear Algebra and Geophysical Migration

Yann-Hervé de Roeck

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Sparse linear algebra and geophysical migration

Yann-Hervé De Roeck

N°3876

Février 2000

THÈME 4

ISSN 0249-6399 ISRN INRIA/RR--3876--FR+ENG

INRIA
RENNES

Sparse linear algebra and geophysical migration

Yann-Hervé De Roeck * †

Thème 4 — Simulation et optimisation
de systèmes complexes
Projet Aladin

Rapport de recherche n ° 3876 — Février 2000 — 52 pages

Abstract: The pre-stack depth migration of reflection seismic data can be expressed, with the framework of waveform inversion, as a linear least squares problem. While defining this operator precisely, additional main characteristics of the forward model, like its huge size, its sparsity and the composition with convolution are detailed. It ends up with a so-called discrete ill-posed problem, whose acceptable solutions have to undergo a regularization procedure. Direct and iterative methods have been implemented with specific attention to the convolution, and then applied on the same data set: a synthetic bidimensional profile of sensible dimensions with some added noise. The efficiency with regard to computational effort, storage requirements and regularizing effect is assessed. From the standpoint of the global inverse problem, the extra feature of providing a solution that can be differentiated with respect to a parameter such as background velocity is also discussed.

Key-words: large sparse linear least squares, geophysical depth migration, deconvolution, regularization, inverse problem, differentiation, QR factorization, conjugate gradient.

(Résumé : tsvp)

* IFREMER, Centre de Brest, BP 70 Technopôle Iroise, 29280 PLOUZANÉ FRANCE
† INRIA/IRISA, Campus de Beaulieu, 35042 RENNES Cedex FRANCE

Unité de recherche INRIA Rennes
IRISA, Campus universitaire de Beaulieu, 35042 RENNES Cedex (France)
Téléphone : 02 99 84 71 00 - International : +33 2 99 84 71 00
Télécopie : 02 99 84 71 71 - International : +33 2 99 84 71 71

Migration géophysique par des méthodes d'algèbre linéaire creuse

Résumé : En se plaçant dans le cadre de l'inversion par formes d'ondes, la migration avant sommation et en profondeur de données de sismique réflexion peut s'exprimer sous la forme d'un problème aux moindres carrés linéaire. Certaines caractéristiques de l'opérateur associé au modèle direct, telles que sa très grande taille, son aspect creux et la composition avec une convolution doivent être prises en compte pour concevoir un algorithme de résolution. Par ailleurs, le très mauvais conditionnement du problème discrétisé nécessite l'emploi d'une procédure de régularisation. Des méthodes directes de factorisation et itératives de type gradient conjugué sont mises en œuvre, en consacrant une attention particulière à l'insertion de la convolution. Un même jeu de données, synthétique, de taille petite mais déjà significative a été sélectionné pour mener ces expériences numériques: il est issu d'un profil bidimensionnel synthétique auquel a été ajouté du bruit. L'efficacité des diverses méthodes est mesurée sous l'angle du coût de calcul, du volume de mémoire nécessaire, et de l'effet régularisant. Du point de vue du problème inverse global que l'on veut résoudre par une méthode d'optimisation locale, l'analyse des méthodes proposées a aussi porté sur la différentiablité des solutions vis-à-vis d'autres paramètres du modèle direct, telle la célérité du milieu.

Mots-clé : moindres carrés linéaires, grands systèmes creux, migration avant sommation, déconvolution, régularisation, problème inverse, différentiation, factorisation QR, gradient conjugué.

1 Introduction

This work aims at enhancing one of the CPU intensive steps of the computations, albeit linear, that arise in geophysical processing of reflection seismic data, namely the pre-stack depth migration [24]. In the first section of this paper, we describe a particular framework for building such an operator, but many other formulations can lead to express this form of inversion by the means of the exact -or approximate- solution of a linear least squares problem [8].

We have therefore paid much attention to the way of solving this very large but sparse and also ill-conditioned linear least-square problem. In geophysical applications, where data and unknown sets are notoriously huge, the issue of sparsity of the migration operator is often not considered, and matrix-free algorithms are chosen without investigating alternative solutions [24], [23], [9]. Nevertheless, the actual computation of the matrix and, if needed, its out-of-core storage extend the range of available algorithms. Namely, on a set of such precomputed matrices, the wide range of numerical experiments conducted in this paper has been made feasible, both by direct and iterative methods. In all cases, a very large rectangular (overdetermined) linear system is to be solved by least squares, but the underlying physics implies characteristic features that any efficient algorithm should take into account.

First, the aforementioned matrices embody the simulation operator or so-called direct model, solution of the wave-equation: hence, it results from the product of the convolution operator built with the source signal by the response of the soil to a Dirac impulse. Therefore, in terms of storage, this multiplication should not be performed: while this remark stands quite obviously for iterative solvers, it also holds for some direct algorithms. The very poor conditioning of the convolution operator carries other harmful impacts: the resulting operator is subsequently more ill-conditioned than the propagation of a Dirac impulse; furthermore, when performing a deconvolution directly on the data, the solution of the least squares problem might differ a lot from what was expected.

Moreover, a foreseeable rank-deficiency appears, due to the fact that the set of points that can be enlightened by the acquisition device, for a given propagation model, is usually smaller than the discretizing grid. In addition, at the boundaries of the enlightened domain, the scarcity of information generates an ill-posed problem. Consequently, the least squares formulation holds all characteristics of a so-called discrete ill-posed problem [2]. We show how this rank-deficiency can be overcome by truncation of direct methods and by the Tikhonov regularization [6], [14]. These techniques have been widely used for another popular geophysical application, namely the tomography [4]. As regards pre-stack depth migration, we report how the techniques allow consistent results to be obtained even at the borders of the illuminated region.

With these characteristics in mind, the pros and cons of direct and iterative methods are discussed. All examples are shown for synthetic data, on a restricted set that mimics marine reflection seismics of high resolution. First on unconvolved and noise-free recordings, then on the more demanding simulation of raw recordings.

On the one hand, today's progress in the efficient implementation of QR factorization [1] suggests that a direct solution will soon be at hand for problems of much bigger size

than our test set. We will stress the ways to overcome storage bulimia: indeed, orthogonal transformations can be performed on the fly, or applied implicitly thanks to the semi-normal equations. Regarding regularization, the best quality results are obtained by truncation [20]. Pivoted algorithms are then required, that inhibit much of the expected efficiency: either huge storage or redundant computations become compulsory. The alternative solution, namely the Tikhonov regularization, does not damage the computational performance of the algorithms designed for sparse full-rank systems, but it is difficult to tune and our displayed results are of lesser quality. Its only advantage, but not the least, consists in providing a solution that can be differentiated with respect to an underlying parameter of the forward model, for instance the background velocity.

On the other hand, the conjugate gradient type algorithms, especially tailored for least squares, provides a reasonable solution [19]. As already experienced in other applications, a well-balanced criterion stops the solution in a filtered state, where the sensitivity due to the smallest singular values has still not occurred. Iterating further would only lead to the need for a regularization tool, of the Tikhonov type for instance. With iterative algorithms, the tuning of the convenient regularization parameters is much faster than with direct factorizations. However, there is not much inference either on the quality of the result nor on the convergence rate. Solutions to speed up the convergence rate, like the search for adequate preconditioners has not been fruitful, but by adapting the ORTHO-CR conjugate gradient algorithms to least squares [5], a remarkable improvement of the convergence rate of the residual has been obtained. However, thanks to the use of synthetic data sets, this study disposes of the actual error with respect to the true solution. In actual practice, a heuristic error estimate, found in the literature [12], slightly tailored to our application, and based upon the residual and some recurrence terms, is shown to be informative for the optimal iteration count. With this new criterion, CGLS happens to be the best algorithm among iterative methods. Moreover, by using the features of a dual method [7], it is possible to employ a slightly modified version of this algorithm in the context of the global non-linear inverse problem.

2 Migration as a linear least squares problem

2.1 A forward model of propagation with diffracting points

Reflection seismic data records the echoes of an acoustic source bouncing back from the soil. When waves encounter discontinuities of elastic moduli (also of density, in some media) between layers, they are scattered and propagated back to the surface. This indirect prospecting technique works especially well in sedimentary areas where most of these surveys take place, mainly for oil exploration. Records that are multi-trace (i.e. on several receivers) and multi-shot (obtained by translating the whole acquisition device), display purposely redundant information. This helps recovering both the reflectors (loci of the aforementioned discontinuities) and the propagation parameters (mainly the velocity of the propagating waves). Figure 1 describes the acquisition principle of a marine 2D survey (with a line of re-

ceivers -a streamer- translating along its axis), while 3D surveys (with an array of receivers) are nowadays routinely performed.

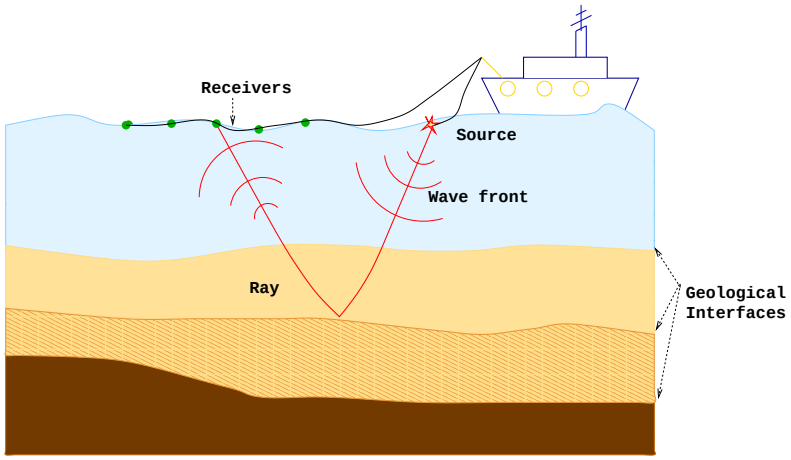


Figure 1: Acquisition principle of a marine 2D survey

Modelling this phenomenon can be expressed in terms of the propagation of elastic waves in a heterogeneous medium, or in a coarser approximation by the propagation of acoustic waves. In this case, the pressure field u which is the observed quantity at the receivers (hydrophones in marine seismics), obeys to the following wave equation due to a pinpoint pressure source term f located at position x_s, therefore the use of the space Dirac function δ :

$$\nu(x)^2 \frac{\partial^2 u(x,t)}{\partial t^2} - \nabla u(x,t) = f(t)\delta(x - x_s)\,. \tag{1}$$

ν is the so called slowness of the wave, inverse of the velocity c, which in turn is proportional to the bulk modulus of the medium. This differential equation holds in this simplified form in a medium of constant density, on assumption made throughout the sequel.

The heterogeneities of the medium, which are expressed by strong variations of $\nu(x)$, tend to scatter the waves. The *first order Born approximation* models this phenomenon by linearizing the wave equation, as stated in [3]. Indeed, the slowness might be decomposed into two terms:

$$\nu(x) = \nu_0(x) + \delta\nu(x)\,,$$

where ν_0 is the reference slowness and $\delta\nu$ the oscillating and small amplitude local variations of ν. If the pressure field is also split into:

$$u(x,t) = u_0(x,t) + \delta u(x,t)\,,$$

retaining the principal quantities and the first order ones in the initial wave equation leads to:

$$\begin{cases} \nu_0^2 \dfrac{\partial^2 u_0}{\partial t^2} - \nabla u_0 = f(t)\delta(x - x_s)\,, \\ \nu_0^2 \dfrac{\partial^2 \delta u}{\partial t^2} - \nabla \delta u = -2\nu_0^2 \dfrac{\delta\nu}{\nu_0} \dfrac{\partial^2 u_0}{\partial t^2}\,. \end{cases}$$

Commonly, the expression of the solution of any of these partial differential equations is simplified by making use of the Green functions. Indeed, a Green function G is the response to an impulse in space and time as right-hand side of the wave equation:

$$\nu(x)^2 \frac{\partial^2 G(x,t;x_s)}{\partial t^2} - \nabla G(x,t;x_s) = \delta(t)\delta(x - x_s)\,.$$

If the source term $f(y,t)$ is an actual distribution in time and in space over the medium $\mathcal{M}$, the solution of the wave equation is given by:

$$u(x,t) = \int_{\mathcal{M}} \int_{-\infty}^{+\infty} G(x,\theta;y) f(y,t-\theta) d\theta\, dy\,,$$

or with the $\star_t$ notation for the convolution in time:

$$u(x,t) = \int_{\mathcal{M}} G(x,.;y) \star_t f(y,.)\, dy\,. \tag{2}$$

In turn, the *high frequency approximation* greatly simplifies the construction of the solutions: if the variations of the reference slowness ν_0 can be considered as smooth with respect to the wavelength λ_f of the propagated signal, i.e.

$$\lambda_f \ll \frac{\nu_0}{\| \nabla \nu_0 \|_2}\,,$$

then the asymptotic form of the Green functions can be used. Hence this approximation is consistent with the previous linearization. In 3D, the asymptotic Green functions can be written as:

$$G(x,t;y) = A(x;y)\delta(t - \tau(x;y))\,,$$

where $\tau(x;y)$ is the travel-time of the propagation between x and y, which obeys to the *eikonal equation*:

$$\| \nabla \tau \|_2^2 = \nu_0^2\,,$$

while $A(x;y)$ is an amplitude factor deduced from the *transport equation*:

$$A\,\Delta\tau + 2(\nabla A)^{\mathrm{T}} \cdot \nabla\tau = 0\,.$$

Again, there are many approximated ways to solve the eikonal and the transport equations. In our developments [16], we solve the eikonal equation in 2D by ray-tracing, integrating an ordinary differential equation along the characteristic lines, with a Runge-Kutta algorithm. Although the resulting travel-time can be a multi-valued function, we have chosen to use a shooting algorithm between each pair of points of a coarse grid, by minimizing the distance of the ray to the targeted point by a Newton algorithm. Then, a linear interpolation on the complete reflectivity grid is made. All together, a single-valued function is computed, which is the proper result in moderately perturbed media.

The amplitude is approximated by the spherical divergence, which makes it equal to:

$$A(x,y) = \frac{1}{2\sqrt{2\pi\nu_0(x)\| x-y \|_2}}\,.$$

In the 2D case also, the asymptotic Green function becomes a distribution, that changes formula (2) into a convolution with a fractional derivative of the source:

$$u(x,t) = \int_{\mathcal{M}} A(x;y)\delta(.-\tau(x;y)) \star_t f^{(-\frac{1}{2})}(y,.)\,dy\,.$$

Hence, for a pinpoint source, the reference pressure field u_0 and the scattered pressure field δu that constitutes the observed data, can be computed by:

$$u_0(x,t) = A(x;x_S)\delta(.-\tau(x;x_S)) \star_t f^{(-\frac{1}{2})}(.)\,,$$

$$\delta u(x,t) = \int_{\mathcal{M}} -2\nu_0(y)^2\frac{\delta\nu(y)}{\nu_0(y)}A(x,y;x_S)\delta(.-\tau(x,y;x_S)) \star_t f(.)\,dy\,, \tag{3}$$

with $A(x,y;x_S) = A(y;x_S)A(x;y)$ and $\tau(x,y;x_S) = \tau(y;x_S) + \tau(x;y)$.

The actual processing of the reflection seismic data then consists in solving the inverse problem provided by equation (3), usually in terms of the following set of variables:

- the *seismograms*, the simulated data, $d(R_i,S_j,t_k)$, that are a very partial observation of the scattered pressure field δu for a collection of locations of receivers x_{R_i} during an interval of discrete recording time, $t_k \in [T_0,T_1]$, while the source is located at locations x_{S_j};
- the *reflectivity*, $r(y) = \dfrac{\delta\nu(y)}{\nu_0(y)}$, a first unknown field, with respect to which equation (3) is linear. From now on, we will write: $d = B.r$, where B is the linear operator mapping the reflectivity r to the observed data d;
- the *background velocity*, $c_0(y) = \dfrac{1}{\nu_0(y)}$, a second unknown field, with respect to which equation (3) is highly non-linear. Hence, there exists a non linear dependency of the modelling operator with respect to the background velocity:

$$d = B(c_0).r\,. \tag{4}$$

We recalled these successive approximations in order first to be precise about the framework in which the solution is sought, and second to highlight the intricate computational steps that constitute this problem.

2.2 Waveform linear inversion

In this paper, we will mostly focus on the linear inversion. That is to say, once a background velocity has been estimated or its field value has been assumed, how the reflectivity field can be recovered. This process called depth migration by geophysicists, namely consists in the search for the location and the amplitude of the reflecting layers. As it results in an interpretive map of the surveyed subsoil, it is considered as an imaging algorithm, among which the most famous is the Kirchhoff algorithm, following the Claerbout imaging principle of diffracting points [9].

However, several authors [23], [7], have shown the link between an approach based on a least-square identification of the reflection seismic data by a wave propagation model (waveform inversion) and this Kirchhoff migration operator. Indeed, the latter amounts to estimate the reflectivity unknown after the first iteration of a conjugate gradient using a proper preconditioner.

Hence, a waveform inversion consists in minimizing by some cost function the norm of the difference between experimental and simulated data. Straightforwardly, the euclidian norm can be chosen,

$$\begin{aligned} J(r;c) &= \| B(c)\, r - d \|_2^2 \\ &= \sum_{s\in\text{shots}} \sum_{h\in\text{receivers}} \sum_{t\in[T_0,T_1]} \left((B(c)\, r)^{s,h}(t) - d^{s,h}(t) \right)^2 , \end{aligned} \tag{5}$$

where d now stands for the observed data (moreover, as the background is supposed to be known, the dependency over c will be dropped in the following). The use of other norms, the weighted versions of the L^2-norm among others, will be discussed later.

The acquisition parameters (receivers, shots, time discretization: the product of the three cardinals of these sets gives the dimension of the data space) and the field model parameters (mesh size and extent of the reflectivity grid) have been sized with respect to the mean frequency of the seismic source. It also leads purposely to an overdetermined problem.

The solution could consist in building the least squares solution, expressed by factorization algorithms (SVD or QR), e.g. by forming the Moore-Penrose pseudo-inverse:

$$r = B^\dagger\, d\,. \tag{6}$$

Up to now, such a direct solution has never been performed in this way for this problem, mainly because the dimensions of the linear operator surge to several millions of rows times several hundreds of thousands of columns. The alternative solution has commonly been first to cast the normal equations (supposing B has full rank):

$$r = (B^{\mathrm{T}}B)^{-1}B^{\mathrm{T}}d \;=\; B^\dagger d\,,$$

then to approximate the solution by:

$$r = K B^{\mathrm{T}} d\,,$$

where K is a coarse approximation of the inverse of $B^{\mathrm{T}}B$. In the many weighted forms that this matrix can take, one can find the Kirchhoff migration [7]. For instance, in our previous work on waveform inversion [17], we have taken $K = \mathrm{diag}(B^{\mathrm{T}}B)^{-1}$ to build such a pre-stack depth migration operator.

Such a choice amounts to multiplying, by a diagonal matrix K, the opposite of the gradient of J in $r = 0$. Thus, it corresponds to the first iteration of a preconditioned gradient method.

Conversely, the so-called *iterative migration* consists in running several iterations of the preconditioned conjugate gradient on the normal equation system. Although quoted by [23], this algorithm has seldom been used.

It is also applied in an alternate form on the time reflectivity s in the Migration Based Travel Time method of inversion [8]. Once a diagonal matrix K has been chosen, a "change of variable" is proceeded by:

$$r = K B^{\mathrm{T}} w\,,$$

and the cost function of definition (5) is then replaced by:

$$\begin{aligned} \mathcal{J}(w;c) &= \| B(c)\, K\, B(c)^{\mathrm{T}} w - d \|_2^2 \\ &= J(r;c)\,. \end{aligned} \tag{7}$$

A square symmetric system has then to be solved, by a conjugate gradient method. As it happens to be a positive, but semi-definite operator, the orthogonal conjugate residual, ORTHO-CR, has been selected for its robustness to this case. We will return later to this algorithm quoted by [5], in Section 5 devoted to iterative methods.

Indeed, in this paper, our goal consists in finding the best approximation of $B^{\dagger}d$, using the most appropriate tools of linear algebra. This could lead to another alternate cost function for the global inverse problem expressed with respect to r and c in the conventional equation (5) or wrt w and c in the MBTT formulation (7):

$$\mathbb{J}(c) = \| B(c)\, B(c)^{\dagger} d - d \|_2^2\,, \tag{8}$$

which is a reduced least squares formulation of the non-linear inversion. The issues of computational efficiency and of accuracy of the estimate of $B^{\dagger}d$ must then be studied very carefully.

3 Sparse storage versus matrix-free

Even if geophysical forward models and migrations generally stand for linear operators, they are usually processed in a matrix-free mode. Indeed, each in their own opposite way, these

operators map variable spaces of huge dimensions, and most often they are applied only once, on only one vector.

Iterative migration makes use of one forward model and one migration (transposed or weighted transposed forward operator) per iteration. Nevertheless, the few implementations of this algorithm remain matrix-free.

However we propose to compute and store the actual matrix of the forward model, because its sparsity remains very high. In Table 1 we show the figures for different problem sizes. The larger the size of the problem, the lower the density of the simulation matrix.

Problem id.	comments	# receivers	# sources	# time samples	# grid steps in x	# grid steps in z	# nonzeros in B	% sparsity
				# rows of B	# columns of B			
1	Dirac synthetics	24	50	100	95	101		
				120 000		**9 595**	**3 953 780**	**0,34%**
1'	Idem, no null column	24	50	100	95	101		
				76 817		**8 286**	**3 953 780**	**0,62%**
1"	Idem, convolved	24	50	100	95	101		
				76 817		**8 286**	**18 312 987**	**2,88%**
2	Dirac synthetics	24	200	400	245	381		
				1 920 000		**93 345**	**117 513 352**	**0,07%**
2'	Idem, no null column	24	200	400	245	381		
				1 920 000		**93 219**	**117 513 352**	**0,07%**
2"	Idem, convolved	24	200	400	245	381		
				1 920 000		**93 219**	**705 079 780**	**0,39%**
3	Real high res., Dirac	24	401	401	842	201		
				3 859 224		**169 242**	**123 536 804**	**0,02%**
3'	Idem, convolved	24	401	401	842	201		
				3 859 224		**169 242**	**988 294 432**	**0,15%**

Table 1: Sparsity of the B matrices for some test models.

3.1 Compressed column format

The columns of the simulation matrix B correspond to the vertices of the 2D-reflectivity grid, hence the total number of columns is the product between the number of grid steps horizontally (in x) and vertically (in z).

Rows of B stand for the seismic data of a multi-trace and multi-shot profile, hence their number amounts to the product of the number of time samples times the number of traces (receivers) times the number of shots (sources). In actual surveys these figures rocket to several millions or billions, but usually, the inversion is performed on restricted domains of interest, therefore it is already meaningful to develop methods entitled to handle subsets of data of a few millions entries.

In practice, the compressed column format has been chosen for this sparse matrix B of nl rows by nc columns, with nnz non-zeros. It consists of [10], [19]:

- one real (double precision) vector Bsp of length nnz containing all the non-zero entries, scanned by columns,
- one integer vector Jline of length nnz containing the row index of all the non-zero entries, in the same order as before,
- one integer vector Jpntr of nc+1 entries, the so-called column pointer, that provides the index in Bsp and Jline of the first entry of each column, the last component being: Jpntr(nc+1)=nnz+1 .

Even neglecting the column pointer, 12 nnz bytes are still necessary. Although the reasons of the inflation of Problem ***3'*** of Table 1 will be explained, this case would require 11.7 Gigabytes to store matrix B ...

Yet on more modest configurations, the matrix barely fits in regular core memories of current computers. It is thus preferable to access records containing columnwise pieces of Bsp and Jline in sequential mode.

We have experienced that the computation and the storage of matrix Bsp is just twice as expensive as one matrix-free matrix vector product, because of its storage on disk. However, the time needed for a matrix-vector product is half as expensive, even when reading the matrix on disk, as compared to the matrix-free version. Hence, whenever more than 4 matrix vector products are involved, without any consideration about storage requirements, the actual computation of the matrix should be preferred, and it would asymptotically divide the evaluation time by 2. Moreover, this strategy allows to apply a much wider scope of linear algorithms for solving our least squares problem.

Problem ***1*** (see Table 1) is taken as the main support for this study. Indeed it involves a "small" 2D set of data to be mapped onto a small reflectivity grid. Factorization and other operations on this B matrix with circa 4 million non-zero entries will fit into the memory of a workstation, and it can be handled for numerical experiments with MATLAB.

Figure 2 displays the pattern of the sparse matrix B at different scales. Although somehow repetitive due to the ordering, this pattern is very irregular, and depends on the background velocity field. This dependency implies that, in the process of solving the global inverse problem, efficient sparse algorithms which utilize a symbolic factorization, will have to rerun this initial step with a new background velocity value.

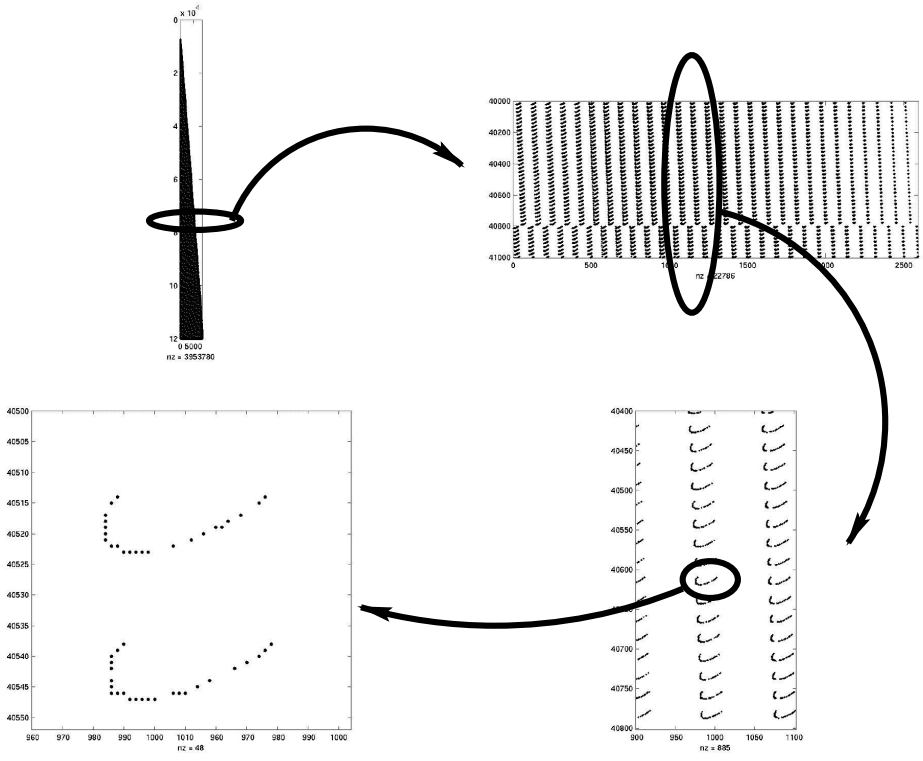


Figure 2: Pattern of the simulation matrix B of Problem ***1'*** (synthetics without convolution), at different levels of detail.

3.2 Dealing with empty columns

Although this is not an issue for sparse computation, the difference between Problems ***1*** and ***1'*** lies in the actual number of non null columns and rows. An empty column of B means that the corresponding vertex of the reflectivity grid does not contribute to the simulation of the recorded data. Indeed, the number of time samples being limited, the reflection on some vertices might propagate too slowly for being actually recorded. Such 'lost' vertices lie on the boundaries of the reflectivity grid. However, they cannot be *a priori* identified before the travel times have been computed. As previously mentioned, the travel-time computation depends on a highly non-linear way from the velocity field. Hence, in the process of the global inverse problem where the velocity field is searched for, the locations, hence the indices, of these null columns steadily vary.

While building the simulation matrix B, our algorithm proceeds by scanning all the vertices of the reflectivity grid (for instance, in the matrix-free implementation, the main

loop is built around the summation over the domain $\mathcal{M}$ of Equation (3)). It is thus easy to detect the empty columns of B, and to provide a reordering of the columns where these latter are flushed to the end. Figure (3) shows a reflectivity field for Problem ***1***. On the left side, it is displayed as originally set up for synthetic simulation. On the right side, one sees the same reflectivity field, however projected on the domain where the columns of B are not empty. The surrounding domain that has been whipped out will thus never be recovered in our inversion process, but one must remember that the shape of this null space depends on the background velocity field. This amounts to a projection of the reflectivity vector r. The number of these null columns equals 9595-8286=1309 for Problem ***1***.

Thus, the number of columns to be discarded might vary a lot, depending not only on the background velocity field but also on the parameters defining the acquisition (number of receivers, of shots, of time samples) and the geometry (size and location of reflectivity grid with respect to the sources and receivers). Hence, on the much larger synthetic Example ***2***, only 126 columns can be discarded instead of 1309 on the small Example ***1***. Even, with a well optimized background velocity field, Example ***3*** on real high resolution data does not lead to any empty column in matrix B.

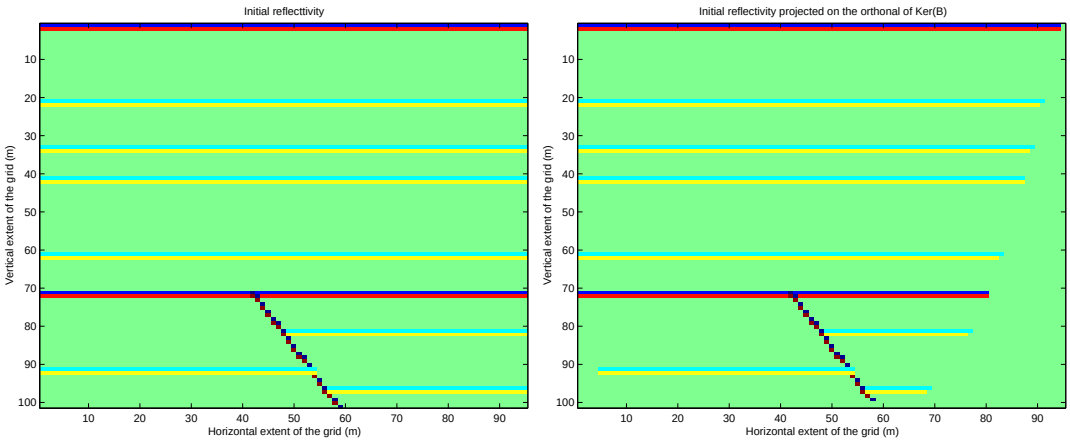


Figure 3: From synthetic example ***1***: on the left side, the reflectivity initially set up; on the right side, the reflectivity at the vertices corresponding to null columns of B has been wiped out. This amounts to a projection of the reflectivity vector r onto a subspace where the least square solution of minimum norm from $\| B \cdot r - d \|_2$ can be found. On this grid, 1309 out of 9595 vertices have been set aside.

Empty rows in B correspond to pieces of seismograms that are always muted. For instance, if the time samples begin with no delay and if the top of the reflectivity grid does not contain the line of receivers, there will never be any information in the first time samples. Because our computational scheme for building B remains columnwise, it is not

easy to detect these empty rows. However, it is not as crucial as for columns, since it does not impact on the rank of the matrix and it does not affect the storage capacity.

3.3 Impact of convolution with the source signal

Yet another issue of sparsity is related to the convolution of the impulse response with the actual source signal propagated in the medium. Indeed, from the wave equation (1) or from formula (3), the linearity with respect to the source signal leads to rewrite[1] the operator defined in (4) by introducing the convolution matrix $C(f)$:

$$
\begin{aligned}
d &= B(c;f).r && (4')\\
&= f * B(c;\delta)\cdot r\,,\\
&= C(f).B(c;\delta)\cdot r\,,\\
\text{with:}\quad \underbrace{C(f)}_{(n_t\cdot n_s\cdot n_r,\, n_t\cdot n_s\cdot n_r)\text{ matrix}} &= P_1 \otimes \underbrace{\begin{pmatrix} f_1 & 0 & \cdots & & \\ f_2 & f_1 & 0 & \cdots & \\ \vdots & \vdots & \ddots & & \\ f_p & \vdots & & & \\ 0 & f_p & & \ddots & \\ & & \ddots & & \\ \vdots & \vdots & & & f_1 \end{pmatrix}}_{(n_t,n_t)\text{ matrix}} \otimes P_2\,, && (9)
\end{aligned}
$$

the source signal being discretized over p samples: $f = [f_1, f_2, \cdots, f_p]$. Permutation matrices, P_1 and P_2, are introduced in order to make clear that the convolution is limited to the range of n_t time samples, while this phenomenon happens for each record, *i.e.* among n_r receivers during n_s shots, whatever the numbering (the data gathering for geophysicists) might be.

Within a matrix-free algorithm, the convolution is performed after the computation of $B(c;\delta).r$, which amounts to convolving with the source signal the impulse response of the medium. Table 1 shows how much fill-in should be expected in matrix $B(c;f)$, for a source signal discretized into $s = 9$ non-zero samples in problems ***1"*** and ***2"***, and $s = 13$ for real data ***3'***. Figure 4 shows the fill-in appearing on problem ***1"***, at the highest level of detail on the pattern of matrix Bsp.

With few assumptions, this fill-in can be predicted. This evaluation is helpful for checking the storage capacities, and for an estimation of the cost of a matrix-vector product.

- the simulation matrix for the impulse response, $B_\delta = B(c;\delta)$, is an $n \times m$ matrix, with N_δ non-zeros;

[1] $*$ is the convolution operator, $\otimes$ is the tensorial product

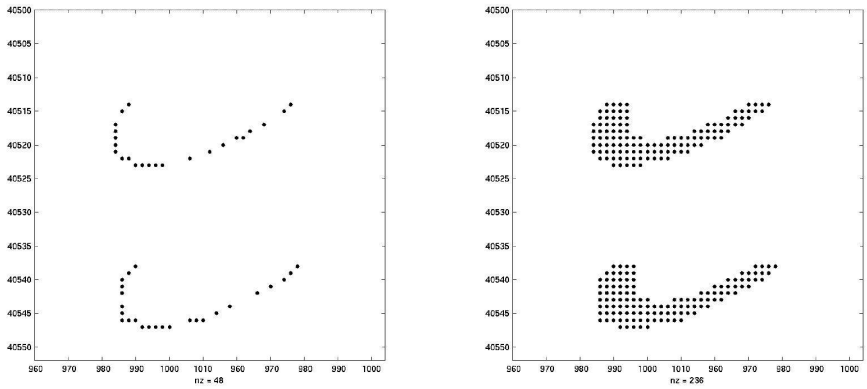


Figure 4: Pattern of the simulation matrix B of problem ***1'*** (synthetics without convolution), and ***1"*** (synthetics with convolution) at the highest level of detail.

- the average fill-in per line, q_δ, is defined by: $q_\delta = N_\delta / n$;
- the non-zero entries of any line of B_δ correspond, for a given pair comprising source, S, and receiver, R, to the set of vertices of the reflectivity grid that lie on an isochrone line for the acoustic propagation (this line is a portion of ellipse of foci S and R, in the case of constant background velocity): see Figure 5. Due to the time discretization scheme, the width (height) of this line, α_δ, is given by the following relation: $(\alpha_\delta - 1)\Delta z = \bar{c}\Delta t$, where Δz is the vertical grid step, Δt the time step, $\bar{c}$ the average background velocity. Usually, Δz and Δt are chosen so that $\Delta z = \bar{c}\Delta t$, thus $\alpha_\delta = 2$;
- an average number of vertices per isochronic can be estimated: $\beta = q_\delta / \alpha_\delta$;
- for the simulation matrix of a regular signal, $B_f = B(c; f)$, with N_f non-zeros, an average fill-in per line $q_f = N_f / n$ can also be defined;
- however, if the propagated source signal has a duration of s time samples, then the width, α_f, of the set of vertices that belong to the isochrones that will interfere during the propagation of the whole signal is linked to the discretization steps by the relation: $(\alpha_f - 1)\Delta z = \bar{c}\, s\, \Delta t$;
- the number of non-zeros of B_f thus becomes:

$$N_f \approx n\alpha_f\beta = n\left(1 + s\frac{\bar{c}\Delta t}{\Delta z}\right)\beta$$

$$\text{while:} \quad N_\delta \approx n\alpha_\delta\beta = n\left(1 + \frac{\bar{c}\Delta t}{\Delta z}\right)\beta$$

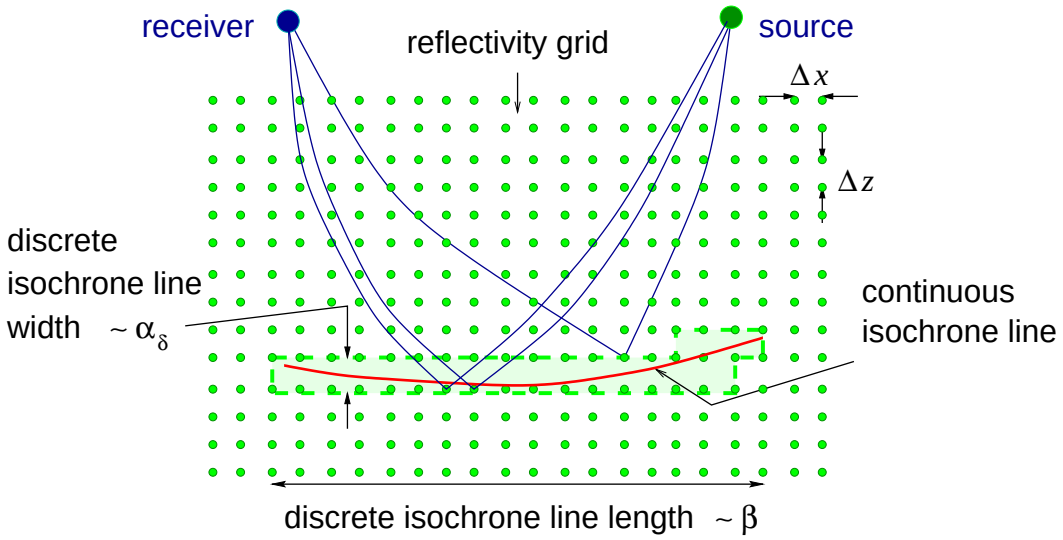


Figure 5: Discretization of isochrone lines: this sketch shows the discrete isochrone line, for which the average width α_δ and length β can be defined. Here, it is assumed that $\Delta z = \bar{c}\Delta t$.

Therefore, in the usual case where $\Delta z = \bar{c}\Delta t$, this estimation becomes:

$$N_f \approx \frac{s+1}{2} N_\delta$$

which is confirmed by the figures of Table 1.

- the number of floating point operations (flop) needed to apply this linear propagation operator to a reflectivity vector is impacted by this fill-in:

$$\begin{aligned} d = B_f \cdot r \text{ costs } \tau_f &= 2N_f \text{ flop} \\ &= 2n\beta(s+1) \text{ flop} \end{aligned}$$

- whereas $C_f = C(f)$, the convolution matrix is also sparse, with an average of s entries per column. Hence:

$$\begin{aligned} d = C_f \cdot (B_\delta \cdot r) \text{ costs } \tau_\delta &= 2N_\delta + 2ns \text{ flop} \\ &= 2n(2\beta + s) \text{ flop} \end{aligned}$$

  τ_δ is then usually much smaller than τ_f, because in general $\beta > s > 2$. Moreover, the cost of building B_f is saved.

Thus, whenever possible, only $B(c;\delta)$ must be stored, and then convolved with the discrete source vector when needed. At first glance, the QR factorization or the building

of a preconditioning matrix by an incomplete factorization seem not to comply with this requirement. Nevertheless, it is possible to design algorithms where the columns of $B(c;\delta)$ are convolved on the fly. Moreover, when the transposed matrix $B(c;f)^{\mathrm{T}}$ is needed, then the transposed operator $C(f)^{\mathrm{T}}$ only requires the knowledge of vector f.

This digression about the implication of convolution in sparse storage also recalls that this multiplication by $C(f)$ renders the problem much more ill-conditioned. The condition number of $C(f)$ depends on the shape of the source signal and exponentially with n_t, the number of time samples in the recording sequence. As the actual matrix $C(f)$ is the result of the aforementioned tensorial products, the number of traces n_r and the number of shots n_s do not impact on the condition number.

Since many deconvolution algorithms exist, one might wonder why keep this expensive and unstable operator in the migration step. Indeed, section 4.7 will show evidence that the data should not be deconvolved before migration, and give insight into the theoretical background for this strategy. In signal processing however, namely of geophysical data [23], it has already been advocated that cross correlation of signals is a more stable operation than deconvolution.

4 Direct methods investigating rank-deficiency and ill-conditioning

As quoted in Section 3.2, a collection of null columns with unpredictable indices (depending on the geometry of the acquisition device and on the velocity field) are easy to detect, and the significative submatrix can be extracted through a reordering.

Still, for the same physical reason, even the latter submatrix has not full rank. Indeed, near locations that are not illuminated and that lead to null columns, some vertices on the reflectivity provide too poor information to the receivers. This rank deficiency can be observed, for instance, while trying to proceed on Problem ***1'***, to a Cholesky factorization of the normal equation matrix $G = B^{\mathrm{T}}B$ (positive semi-definite matrix): a null or slightly negative pivot is encountered. Moreover, in perfect arithmetic, the rank of the normal equation matrix G is the same as the one of B, but it can be expected to be lower numerically, due to the finite representation of floating point numbers while dealing with an average of squared coefficients.

Nevertheless, the true rank of B is not the issue. Indeed, the error analysis of the solution of a perturbed least squares problem says more about the concept of numerical rank which needs to be taken into consideration, and which leads to building a low-rank approximation to B.

4.1 Theoretical insight through the SVD decomposition

B, as any general matrix, can be factorized by a singular value decomposition (SVD), for $m > n$, [11]:

$$\underbrace{B}_{(m,n)} = \underbrace{U}_{(m,m)} \cdot \underbrace{D}_{(m,n)} \cdot \underbrace{V^{\mathrm{T}}}_{(n,n)} ,$$

$$\text{with orthogonal matrices:} \quad \begin{cases} U^{\mathrm{T}} \cdot U &= I_m \\ V^{\mathrm{T}} \cdot V &= I_n \end{cases}$$

$$\text{and diagonal rectangular matrix:} \quad D = \mathrm{diag}(\underbrace{\sigma_1, \sigma_2, \cdots, \sigma_q}_{\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_q > 0}, 0, \cdots, 0) , \qquad (10)$$

$$\text{with } q = \mathrm{rank}(B) \leq n .$$

With this decomposition, the 2-norm of B lies in σ_1, the largest of all *singular values* $\{\sigma_i\}_{i \in [1:q]}$. When B has a rank q lower than n, the least squares problem induces a whole variety of solutions (of direction $\mathcal{N}(B)$, the kernel of B): usually, the solution of minimum norm is chosen. It is obtained as in (6) by multiplying the right hand side (data d) by the Moore-Penrose pseudo inverse, $B^{\dagger}$, which is defined by

$$\underbrace{B^{\dagger}}_{(n,m)} = \underbrace{V}_{(n,n)} \cdot \underbrace{D^{\dagger}}_{(n,m)} \cdot \underbrace{U^{\mathrm{T}}}_{(m,m)} ,$$

$$\text{where } D^{\dagger} = \mathrm{diag}(\frac{1}{\sigma_1}, \frac{1}{\sigma_2}, \cdots, \frac{1}{\sigma_q}, 0, \cdots, 0) .$$

Consequently, $\frac{1}{\sigma_q}$ becomes the 2-norm of $B^{\dagger}$. A generalized definition of the 2-norm condition number is inferred:

$$\kappa_2(B) = \frac{\sigma_1}{\sigma_q} .$$

Let us now consider a perturbed system: for instance, perturbations are due to some inescapable noise δd on the data and to the approximations made on our model for the linear operator, δB. Then, (if $\| \delta B \|_2 \leq \sigma_q$), according to [2], the first order error analysis gives:

$$\text{let } \bar{r} \text{ be the minimum norm solution to:} \quad \min \| B \cdot r - d \|_2 ,$$

$$\bar{r} + \delta r \text{ the minimum norm solution to:} \quad \min \| (B + \delta B) \cdot r - (d + \delta d) \|_2 ,$$

then

$$\| \delta r \|_2 \leq \frac{1}{\sigma_q} \left(\| \delta d_{\mathcal{R}(B)} \|_2 + \| \delta B \|_2 \| r \|_2 \right) + \frac{1}{\sigma_q^2} \| \delta B \|_2 \| B \cdot r - d \|_2 ,$$

where $\delta d_{\mathcal{R}(B)}$ is the projection of the perturbation of the data on the image (range) of B.

Obviously, the smaller σ_q, the larger the error. Thereby, a truncated SVD (TSVD) solution is provided to such problems, [2]: among the singular values, a numerical threshold is set under which they are all considered to be null. This is equivalent to looking for the pseudo inverse of an approximate matrix $\tilde{B}$ which has a prescribed lower rank than B:

$$\begin{aligned} \tilde{B} &= U\tilde{D}V^{\mathrm{T}}\,, \\ \text{where } \tilde{D} &= \mathrm{diag}(\sigma_1, \sigma_2, \cdots, \sigma_p, 0, \cdots)\,, \\ \text{with } p &< q\,. \end{aligned}$$

$\tilde{B}$ is proved to be the best approximation to B among matrices of rank p, and moreover, on the matrix norm again, a bound is given:

$$\| B - \tilde{B} \|_2 \leq \sigma_p\,.$$

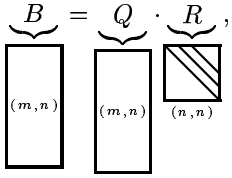
It must be stressed that truncation or other means of regularization happens to be compulsory to obtain acceptable results out of any ill-conditioned least squares problem. Otherwise, the action of the inverse of the smallest singular values inflates the effects on the perturbations of the system.

4.2 Sparse QR with no pivoting

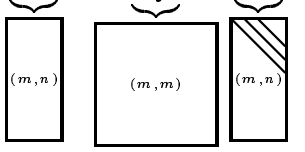
In spite of these nice properties, TSVD requires huge storage capacities due to fill-in in the orthogonal factor, and undergoes a prohibitive computational cost, as it can be connected to an eigenvalue problem (that of the normal equation matrix). The QR factorization can be adapted to solve rank deficient least squares problems efficiently, at the cost of another layer of approximations.

Two types of QR algorithms are acknowledged. In the case of an underlying Gram-Schmidt (GS) or Modified Gram-Schmidt (MGS) orthogonalization algorithm, Q is an (m,n) orthogonal matrix and R is (n,n) square upper triangular. In case of a factorization through Householder transforms (HT), i.e. by successive orthogonal symmetries, or Givens transforms (RT), i.e. by rotations, Q is then (m,m) square and orthogonal while R is (m,n) rectangular, but still upper triangular. Indeed, these algorithms are often quoted as:

the *thin* [M]GS algorithms



and the *fat* [H,R]T algorithms



The *fat* case can however be squeezed into the *thin* one, after computation:

$$\underbrace{B}_{(m,n)} = \underbrace{\begin{bmatrix} Q^1 & Q^2 \end{bmatrix}}_{\substack{(m,n) \quad (m,s) \\ s=m-n}} \cdot \underbrace{\begin{bmatrix} R^1 \\ 0 \end{bmatrix}} = Q^1 \cdot R^1 \,.$$

However, in the factorization process of the *fat* algorithm, either Q^2 is stored (although some economical forms exist), or the right-most columns of R are filled in, up to the very last step. This becomes a very crucial point when m is very large, as in our case.

Let us then consider the *fat* formulation, as it leads to the simplest notations. The key point of the solution of the least squares problem by QR arises from the isometric property of the orthogonal matrices:

$$\begin{aligned} \| B \cdot r - d \|^2_{2,\mathbb{R}^m} &= \| Q \cdot R \cdot r - d \|^2_{2,\mathbb{R}^m} \,, \\ &= \| R_1 \cdot r - Q_1^{\mathrm{T}} \cdot d \|^2_{2,\mathbb{R}^n} + \| Q_2^{\mathrm{T}} \cdot d \|^2_{2,\mathbb{R}^s} \,. \end{aligned}$$

A straightforward solution can then be extracted if and only if R_1 is non singular (it has no zeros on its diagonal). Then :

$$\text{solve } R_1 \cdot \bar{r} = Q_1^{\mathrm{T}} d \,.$$

Some very efficient implementations of this QR factorization have been developed for sparse matrices, however in this simple form without column pivoting. The numerical interest of the pivoting process being discussed further, the key point here consists in obtaining a manageable and efficient factorization of the matrix.

For the purpose of solving least squares problems, these implementations deliver on the fly the product $Q^{\mathrm{T}} y$ of a provided right-hand side y, hence no storage is required for Q. Anyhow, these methods are often quoted as Q-less QR, solely providing the R factor. Indeed, in the case R is of full rank, Q is considered to be equal to:

$$Q_1 = B \cdot R_1^{-1} \,,$$

then the semi-normal equations can be used to form the solution of the least squares problem:

$$\begin{aligned} \min_r \| B \cdot r - d \|_2 \Rightarrow & \\ r &= R_1^{-1} \cdot Q_1^{\mathrm{T}} \cdot d && \text{(practically, solve } R_1 \cdot r = Q_1^{\mathrm{T}} \cdot d) \\ &= R_1^{-1} \cdot R_1^{-T} \cdot B^{\mathrm{T}} \cdot d && \text{(solve first } R_1^{\mathrm{T}} \cdot y = B^{\mathrm{T}} \cdot d \\ & && \quad \text{then solve } R_1 \cdot r = y) \,. \end{aligned}$$

Such a sparse version exists in MATLAB, and this is the only way to handle the seismic simulation matrix B, even with the dimensions of problem ***1*** of Table 1. For a practical

use on our problems of standard size (e.g., problem ***2*** or ***3***), there are some very efficient parallel FORTRAN codes based on a multi-frontal approach [1]. There are still two difficulties to overcome when using these powerful techniques: on the one hand, no pivoting is available, since it is easy to understand why column interchange stands as an impediment with respect to storage and computational efficiency; on the other hand, a specific implementation would be necessary in order to proceed to the convolution on the fly, column by column, thus avoiding the full storage of the initial matrix.

On the resulting R factor, the return time and the sparsity pattern depend on the initial column ordering of B. We have tried a column minimum degree (COLMMD), which is supposed to lead to small fill-in, and a simple column and row reverse permutation (CLREV), which symmetrically transforms the pattern of B from lower to upper "triangular" (wrt. the diagonal of the rectangle, not the usual principal diagonal, as it can be observed on Figure 2 at the coarser level of detail, the upper left subplot).

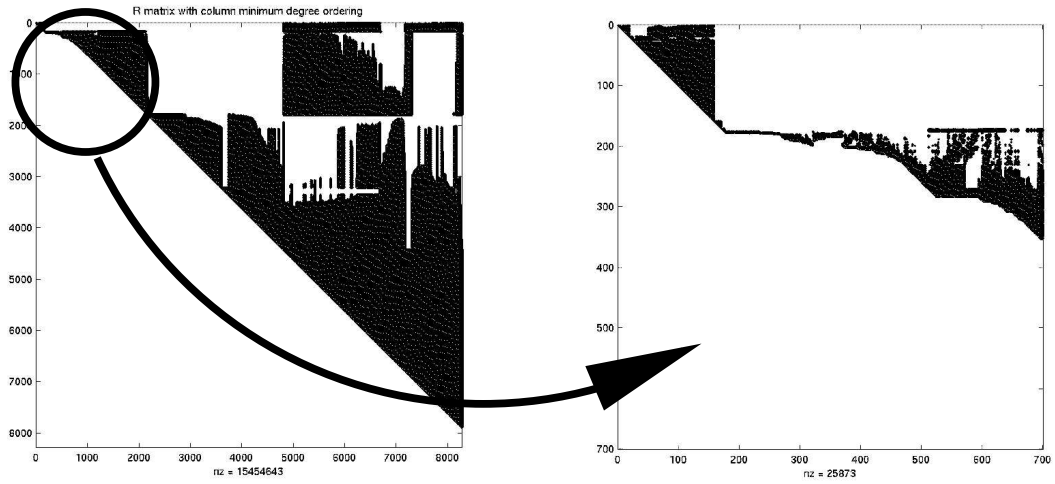


Figure 6: R factor pattern. Matrix B had no null column, and its columns have been permuted by the minimum degree algorithm. The zoom on the upper diagonal block shows that null pivots are encountered. Thus there is a rank deficiency.

However, as expected in our case, the R factor displays a pattern with zeros on the main diagonal, as shown on Figure 6. R is thus singular, and cannot be used for solving the least squares problem. Moreover, even if it is possible to get a hint of the rank of the system from the distance between the diagonal and the non-zero coefficients, see Figure 7, this statement holds in perfect arithmetic only. A different dimension of the kernel is estimated for two re-orderings: 390 with a column minimum degree COLMMD, 386 with CLREV. There are still some approximations that can be made directly on this factorization without pivoting, by manipulating the R factor: firstly, from top to bottom, only the columns whose non-

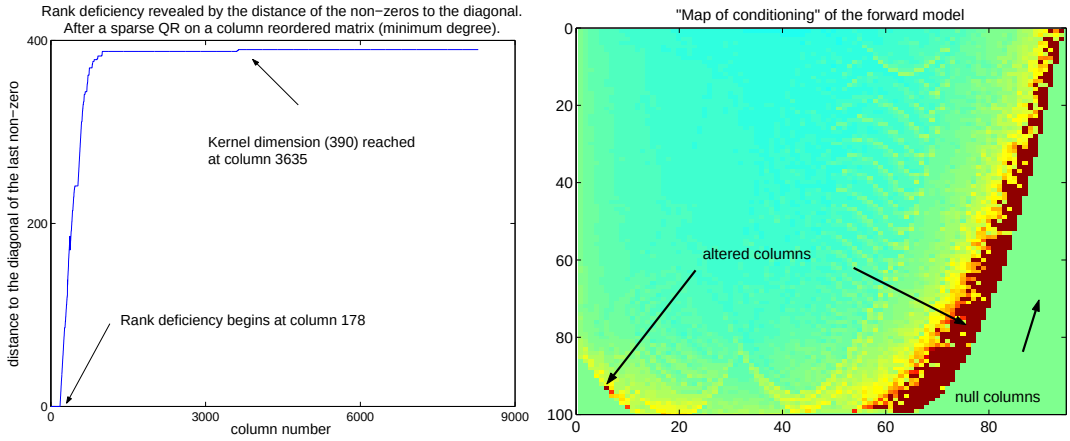
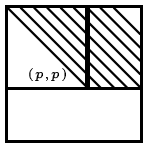


Figure 7: *left*: Distance to the diagonal term of the last non-zero element of each column. Each jump towards the top detects a new singularity.
right: Displayed on the reflectivity grid, the relative amplitude of the last non-zero element of each column of R. Dark locations stand for each column on which a new kernel direction has been detected.

zeros hit a new row are selected, as they correspond to vectors that expand the range of the matrix[2]. When p such columns are selected in R to the left, the other ones being expelled to the right, a column permutation is obtained, after which R has the following shape:

$$R = \begin{bmatrix} R' & R" \\ 0 & 0 \end{bmatrix} = \quad , \tag{11}$$



with R' full rank p upper triangular matrix $(p < n)$,
and $R"$ a rectangular $(p, n-p)$ matrix.

$$\text{Then, } R \cdot x = R \cdot \begin{pmatrix} x' \\ x" \end{pmatrix} = \begin{pmatrix} y' \\ y" \end{pmatrix},$$

[2]Indeed, columns of B are linear combinations of columns of Q (following notation: one subscript for column vectors, 2 subscripts for scalar coefficients):

$$\forall j \in [1:n],\ B_j = \sum_{i=1}^{j_0, j_0 <= j} R_{ij} Q_i .$$

Hence, for any given j, B_j adds a new direction to the span of the preceding columns of B if and only if, j_0 being the index of the last non null row coefficient of column vector R^j, then $\forall k < j$, $R_{j_0 k} = 0$.

is an underdetermined system decomposed on subspaces of respective size p, $n-p$. It would have a proper minimum norm solution, but usually the so-called *basic* solution is extracted:

$$\begin{cases} \text{solve } R' \cdot x' = y' \,, \\ \text{set } x" = 0 \,. \end{cases}$$

Unfortunately, the result is not satisfactory, due to the very bad conditioning of matrix R'. Indeed, this permuted (not yet pivoted) version of QR has kept all the singular values of B in R', even though the previous section has shown the need for a truncation.

The physical meaning of rank deficiency is confirmed, when displaying the location of the missing columns of R' on the reflectivity grid. The migration problem seems obviously ill-posed on the boundaries of the domain, close to where null columns can be found (see Figure 7). These locations correspond to the loci where common geophysical processing display the so-called migration smiles.

In turn, additional columns of R' might be discarded in order to obtain a better conditioning. This can be performed while keeping the sparse QR factorization, by applying a sequence of Givens rotations to zero the coefficients arising under the diagonal (due to the shift to the left of the remaining columns). Still, what is the threshold for discarding such or such column? The normalized absolute value of the pivot (the diagonal coefficient normalized by the largest entry of the column) can be chosen, as this is the source of the inflation of the perturbation. But this technique has not proven to be successful in our tests.

4.3 Regularization by Truncated Pivoted QR methods

The pivoted QR factorization (PQR), proceeds by column pivoting at each step of the factorization, in order to obtain an R factor whose entries are steadily decreasing. This factorization leads to:

$$B \cdot E = Q \cdot R$$

with E a permutation matrix, Q an orthogonal matrix and R an upper triangular matrix.

The permutation E can be stored economically as a vector, and be used in the following way:

$$B(:, E) = Q \cdot R \qquad \text{(in MATLAB notations)}$$

The various PQR algorithms lead to an R factor whose diagonal coefficients have decreasing absolute values. They usually prescribe that :

$$\forall i \in [1:n],\ |R_{ii}| \geq \| R_{[i:n,j]} \|_2, \forall j \geq i+1 \,. \tag{12}$$

That is to say, each pivot is larger than the euclidian norm of any remaining sub-column during the factorization and at the completion of it. This hypothesis (12) implies that,

$\forall i \in [1:n]$:

$$
\begin{aligned}
(n-i+1)\,|R_{ii}|^2 &\geq \sum_{j=i+1}^{n} \| R_{[i+1:n,j]} \|_2^2 \,, \\
&\geq \| R_{[i+1:n,i+1:n]} \|_F^2 \,, \text{ (Frobenius norm)} \\
&\geq \| R_{[i+1:n,i+1:n]} \|_2^2 \,, \text{ (majoration of induced matrix 2-norm)} \\
\Rightarrow \forall i \in [1:n], & \\
\| R_{[i+1:n,i+1:n]} \|_2 &\leq \sqrt{n-i+1}|R_{ii}| \,. \qquad (13)
\end{aligned}
$$

This bound allows to neglect the lower block diagonal term in R after a given threshold, hence to retrieve a shape similar to the one displayed on Equation (11), from which a *basic* solution can be evaluated. However, for the remaining upper diagonal block R', of size p, it can only be proven that:

$$
\begin{aligned}
\kappa_2(R') &= \frac{\sigma_1(R')}{\sigma_p(R')} \,, \\
&\geq \frac{|R_{11}|}{|R_{pp}|} \,.
\end{aligned}
$$

This inequality involves the singular values of R', and R being an extension of R', the following inequalities hold [2]:

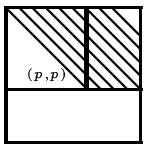
$$
\begin{aligned}
\sigma_1(R) \geq \sigma_1(R') \;&\geq\; |R_{11}| \,, \\
\sigma_p(R) &\geq \sigma_p(R') \,, \\
|R_{pp}| &\geq \sigma_p(R') \,. \\
&\text{Hence, if: } |R_{pp}| < \delta |R_{11}| \,, \\
\| R_{[p+1:n,p+1:n]} \|_2 &\leq \delta\sqrt{n-p+1}\,\sigma_1(R) \,.
\end{aligned}
$$

Therefore, with such a criterion, the neglected part of the matrix is bounded (no important contribution has been left aside), but there is no theoretical evidence that R' has a convenient condition number. Some counter-examples have already been shown, [2]. However, in practice, for a PQR algorithm respecting (12), the order of magnitude of $|R_{ii}|$ has been experienced to be close to $\sigma_i(R)$ for random matrices.

Hence splitting R with a threshold of the kind:

$$
\forall i \in [1:p] \,, \; \frac{|R_{ii}|}{|R_{11}|} \geq \delta \qquad (14)
$$

(15)

assumes an R factor of the shape: 

, (16)

that should approximate the TSVD keeping all singular values $\sigma_i > \delta\sigma_1$. This algorithm is called truncated pivoted QR (TPQR).

However, PQR only exists in full matrix form within MATLAB, while in our test case, B as a full matrix would have 994 320 000 entries (*circa* 8 Gigabytes would be necessary...). Section 4.5 shows how to reduce the problem size on which TPQR can be used in full matrix form. However, Section 4.6 displays an algorithm as well as some adaptations, leading to a sparse processing that is better suited to our application.

Pivoted QR has also been modified to be precisely rank-revealing [6], [18], but inevitably, an iterative procedure is then involved: the heuristic advises 2 or 3 iterations only of inverse power algorithm to find out the smallest singular value of R, thanks to iterated call to forward and backward substitution. The implied overcost has prevented us from trying out this method on our problem.

4.4 Regularization of the convolution operator

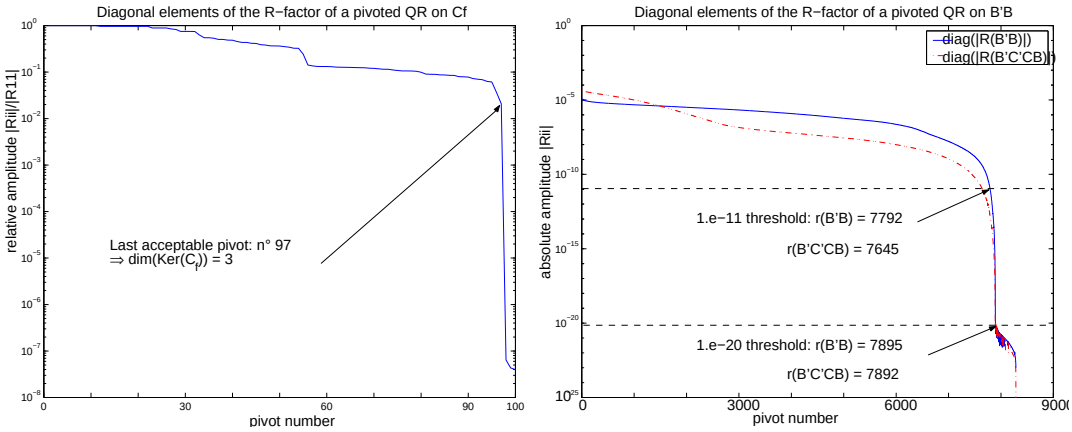


Figure 8: *left*: Display of the diagonal elements of the R factor of pivoted QR factorizations, on the convolution operator C_f.
right: Display of the diagonal elements of the R factor of pivoted QR factorizations: on the normal equations, $G = B^{\mathrm{T}}B$ for the plain curve, $G_f = B_f^{\mathrm{T}}B_f$ (*i.e.* with a convolution) for the dashed line.

On the convolution operator C_f, thanks to the tensorial product observed on equation (9), the PQR factorization can be performed on a "small" (n_t, n_t) matrix, and can therefore be handled as dense. Figure 8 shows the abrupt decline of the absolute value of the diagonal coefficients of the R factor. The numerical rank of this operator is thus easy to determine: in this particular case, the dimension of the numerical kernel is 3.

Experiments on deconvolution of noisy data with a truncated pseudo-inverse of this rank are very satisfactory. However forthcoming arguments will show that it might be useless to build such a deconvolution operator (to be noted $\tilde{C}_f^{\dagger}$ in the following).

4.5 Usage of full TPQR: on normal equations and on a not pivoted R factor

One solution to reduce the problem size (hence to be able to handle full matrices) consists in using the normal equations $G = B^{\mathrm{T}}B$. The numerical rank is certainly lowered by this procedure, because the singular values of G are the squares of those of B, but at least G can be manipulated as a full matrix. The density of this matrix is also very high, especially in our small test case ***1'***. Indeed, without convolution, G has $16.\ 10^6$ non-zeros, which amounts to a density of 24%; with convolution, $G_f = B^{\mathrm{T}}C_f^{\mathrm{T}}C_fB$ has $23.\ 10^6$ non-zeros and its density reaches 34%. The density of the normal equation matrix is logically less influenced by the insertion of the convolution operator.

The decrease of the absolute values of the diagonal elements of the R factor from PQR factorizations is smoother on the propagation operators, as shown on Figure 8, than on the previous convolution operator. This shape expresses, throughout the aforementioned approximations, the decrease of the singular values. The observed smoothness implies that the propagation operators are discrete ill-posed problems, according to customary denomination [14].

The combined operator (convolution after propagation, $C_f \cdot B$, the dotted line on Figure 8) shows an even smoother behaviour than the propagation of the sole Dirac signal (the plain line). Hence it will be more difficult to extract a satisfying threshold for the truncation.

When the ratio $|R_{ii}|/|R_{11}|$ draws nearer to the machine precision, numerical instabilities become perceptible, producing the scrambling of the ends of both curves below a relative value of 10^{-15}.

Another way of processing with full matrix code consists of performing a PQR on the R factor previously obtained without pivoting, by sparse techniques. This R factor, with the same dimensions as normal equations but retaining a better conditioning, can in turn be operated as a full matrix. Thus:

$$\left\{ \begin{array}{lcll} B & = & Q_1 \cdot R_1 & \text{sparse QR} \\ R_1 \cdot E_2 & = & Q_2 \cdot R_2 & \text{full PQR} \end{array} \right. \Rightarrow B \cdot E_2 = Q_1 \cdot Q_2 \cdot R_2 \,. \qquad (17)$$

With storage economy in sight, a Q-less form is kept, i.e., solely R_2 and the permutation vector corresponding to E_2.

This more precise computation leads to actual zeros on the diagonal of the R factor, which displays the adequate shape for the basic solution of equation (16). It follows a dimension of 386 for the kernel of B in problem ***1'*** (+1309 null columns wrt. problem ***1***).

Results displayed on Section 4.7, when only B_δ is factorized by TPQR, are obtained by this method.

4.6 Quasi Gram Schmidt TPQR algorithm

Several TPQR algorithms are proposed by Stewart [20], but only one can be adapted to be performed in an economical Q-less mode, thanks to a downdating technique while performing a modified Gram-Schmidt orhogonalization. This algorithm is reproduced hereafter, as it is the most convenient to apply to our problem. Permutations are implicitly performed by using a permutation vector, which is one of the outputs of the algorithm, together with the numerical rank and the R factor.

Input: a matrix B of dimensions (n, m), an order of truncation $\eta = 10^{-\delta}$.
1: Initialisation: permutation matrix $E = [1 : m]$; iteration count $k = 1$; etc., ...
2: $\nu_j = \| B[:, j] \|_2^2,\ j = [1 : p]$ {computation of the column norms}
3: Extract $\nu_{p_1} = \max_j (\nu_j)$ and store $\varepsilon = \eta \nu_{p_1}$;
4: **while** $\nu_{p_k} \geq \varepsilon$ and $k \leq m$ **do**
5: $E(k) \rightleftarrows E(p_k)$ {column permutation}
6: $v = B[:, E(1 : k-1)]^{\mathrm{T}} * B[:, E(k)]$ {beginning of implicit projection}
7: solve $R[1 : k-1, 1 : k-1]^{\mathrm{T}} * R[1 : k-1, k] = v$
{implicitly, $R[1 : k-1, k] = Q[:, 1 : k-1]^{\mathrm{T}} B[:, k]$}
8: solve $R[1 : k-1, 1 : k-1] * w = R[1 : k-1, k]$
9: $q = B[:, E(k)] - B[:, E(1 : k-1)]$
{implicitly, $q = (I - Q[:, 1 : k-1]^{\mathrm{T}} Q[:, 1 : k-1])\ B[:, k]$}
{**here, re-orthogonalization steps could be added.**}
10: $R[k, k] = \| q \|_2$
11: $q = q / R[k, k]$ {normalization}
12: $r[k+1, m] = q^{\mathrm{T}} B[:, E(k+1 : m)]$
13: $\nu_j = \nu_j - r[j]^2,\ j = [1 : p]$ {update of the column norms}
14: extract $\nu_{p_{k+1}} = \max_{j>k} (\nu_j)$ {search for maximal pivot}
15: $k \hookleftarrow k + 1$
16: **end while**
Output: $k = k - 1$ {numerical rank is $m - k$}, $R[1 : k, 1 : k]$ and $E[1, m]$.

Algorithm 1: Truncated Pivoted QR factorisation: Quasi-Gram-Schmidt version, without reorthogonalization (TPQR-QGS).

The price to pay arises while downdating with an increasing R factor, which might become unaffordable. Indeed, the orthogonalization steps are performed with an implicit Q factor, implying each time one forward and one backward substitution with the R factor. This computational cost might be doubled if reorthogonalization is required (steps to be inserted between 9. and 10.). However, the fact that the computation is stopped after reaching a given truncation threshold avoids most needs for re-orthogonalization.

If the updated column norms are saved when the algorithm breaks down, it is then easy to restart it. Otherwise, these norms can be recomputed by using the following formulae

(using triangular solvers on matrix right-hand side):

$$\nu_j = R_{jj}^2, \quad j = [1:k] \; ;$$

$$\begin{aligned} Y &= B[:, E[1:k]]^{\mathrm{T}} * B[:, E[k+1:m]] \; ; \\ \text{solve} \quad & R[1:k-1, 1:k-1]^{\mathrm{T}} * Z \; = \; Y \; ; \\ \text{solve} \quad & R[1:k-1, 1:k-1] * W \; = \; Z \; ; \\ \nu_j &= \| \, B[:, E[j]] - Z_j \, \|_2^2, \quad j = [k+1:m] \, . \end{aligned}$$

These instructions can also replace step 13 of the algorithm, when the updates on the column norm lead to negative values. On our test case, this happens when ν_j/ν_1 goes below 10^{-18}, hence only rounding errors seem to be involved. Let us however remember that this ratio corresponds to $|R_{ii}/R_{11}| \approx 10^{-9}$, which will be considered in the sequel as an overwhelming stopping criterion for TPQR.

Above all, it must be stressed that this algorithm is well adapted for inserting a convolution operator:

- step 6 becomes: $v = B[:, E(1:k-1)]^{\mathrm{T}} * (C_f^{\mathrm{T}} * (C_f * B[:, E(k)]))$;
- step 9 becomes: $q = C_f * (B[:, E(k)] - B[:, E(1:k-1)])$;
- step 12 becomes: $r[k+1, m] = (C_f^{\mathrm{T}} * q)^{\mathrm{T}} B[:, E(k+1:m)]$.

On our test case, keeping the convolution in this stage rather than computing the combined matrix B_f speeds up by 3 the factorization.

This form of TPQR algorithm has been used with success on our problem without any initial reordering. It provides the most significant results shown in Section 4.7.

4.7 Results with TPQR on different orders of truncation

The data sets are produced by the direct model, $d = B_f \cdot r = C_f \cdot B \cdot r$. However, to avoid any unfounded optimism, some noise must be added to synthetic data, in order to have a right-hand side that does not belong to the image of the operator, as in reality (remind the influence of δd in the perturbation analysis). The synthetic data is generated by the following procedure:

- set r_0 be a given reflectivity, the same as displayed on Figure 3;
- compute noise-free data: $d \; = \; B \cdot r_0$;
- compute noisy data: $d \; = \; B \cdot r_0 + \epsilon$, each component of ϵ being uniformly distributed in the range $[-\tau, \tau]$ with $\tau = 0.05 \| \, B r_0 \, \|_2$;

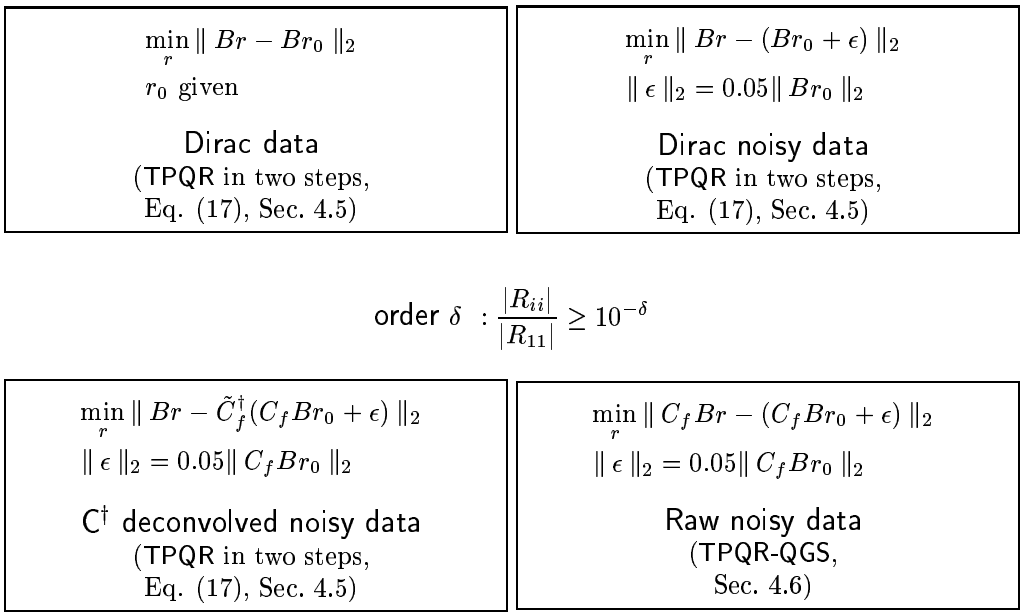


Figure 9: Sketch of the following fourfold figures.

- for convolved data, equivalently: $d_f = C_f \cdot B \cdot r_0 + \epsilon$, each component of ϵ being uniformly distributed in the range $[-\tau, \tau]$ with $\tau = 0.05 \| C_f Br_0 \|_2$.

TPQR is set with the criterion: $|R_{ii}|/|R_{11}| \geq 10^{-\delta}$, δ being called the *order* of truncation, and we have tried to optimize the rendering of the reflectivity (is $\bar{r}$ close enough to r_0?) on integer values of δ only.

On Figure 9, the sketch of the following figures is explained. Results are displayed on a fourfold frame: for noise free Dirac data (top left), noisy Dirac data (top right), noisy but *ab initio* deconvolved data (bottom left), noisy convoluted data (bottom right).

The bottom left frame needs further explanations. Indeed, once the combined operator is used ($C_f \cdot B$), one might be tempted to perform the deconvolution *ab initio*, before the migration. This frame shows the results of this strategy, with a deconvolution performed with the TPQR of operator C_f. However, it must be pointed out that, against all odds:

$$(C_f \cdot B_\delta)^\dagger \neq B_\delta^\dagger \cdot C_f^\dagger \, .$$

Indeed, the equality is false in theory, because:

$$\mathcal{R}(B_\delta) \not\subset \mathcal{N}(C_f)^\perp \, .$$

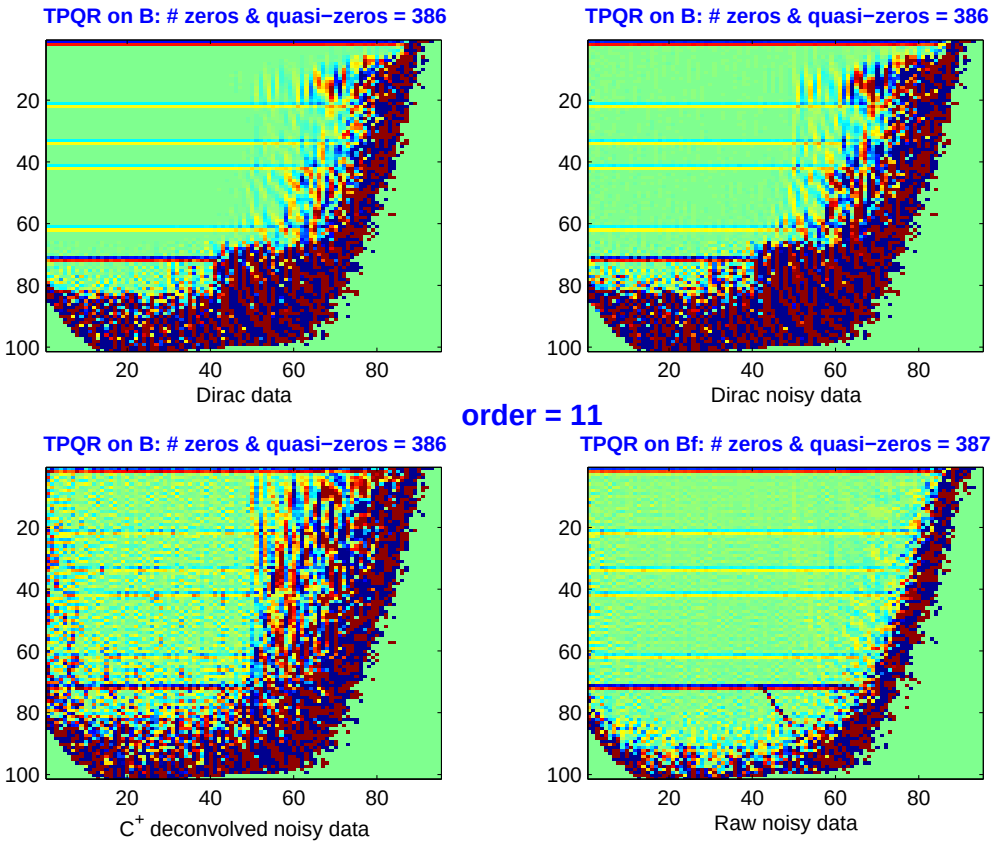


Figure 10: Solutions with almost no truncation ($\delta = 11$ on B but $\delta = 9$ on B_f, due to a breakdown of TPQR-QGS on negative column norms).

A sufficient condition would be that:

$$\text{rank}(B_\delta) = \text{rank}(C_f) \tag{18}$$

$$\text{and } C_f \text{ full rank.} \tag{19}$$

In practice, our operators do not interact that way. B represents a very overdetermined system, so equality (18) is far from being realized. Moreover, the use of a truncated deconvolution operator $\tilde{C}_f^\dagger$ implies that condition (19) is not satisfied. Consequently, the results displayed in the bottom left-hand corner are expectedly bad, whatever the order of truncation on B.

In the four cases, the results obtained with a very small truncation (order 11, Figure 10) are blurred on the fringe. This again proves that the discrete model is ill-posed on the border

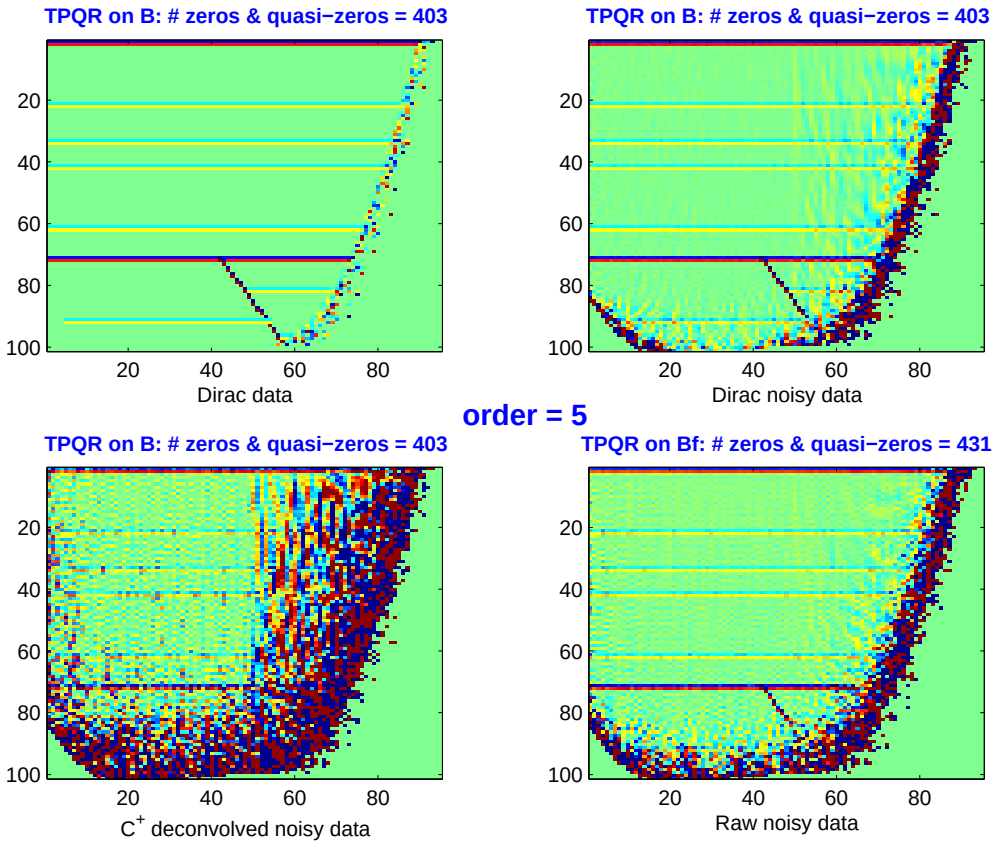


Figure 11: Optimal truncation ($\delta = 5$) for noise free Dirac data.

of the reflectivity domain: insufficient illumination of the sub-soil by the acquisition device is the physical counterpart of this remark.

When order 5 is selected (Figure 11), noise-free data is recovered at its best (top left-hand corner). Lower orders will just add inaccuracy all over the reflectivity field (remember that matrix B is approximated at a lower rank, and that the dimension of the numerical kernel, which is quoted at the top of each plot, logically increases when order decreases).

An optimal order of 2 handles noisy data with the narrowest blurred fringe (both right-hand subplots on Figure 12). There is no hope for the bottom left-hand subplot, where deconvolution has been performed too soon. The bottom right-hand subplot does not reveal as much information as the top right-hand one: convolution has expectedly removed part of information. Choosing $\delta = 2$ implies that the TPQR-QGS algorithm could have been stopped quite early, once 82% of the columns of the matrix have been factorized. As columns are

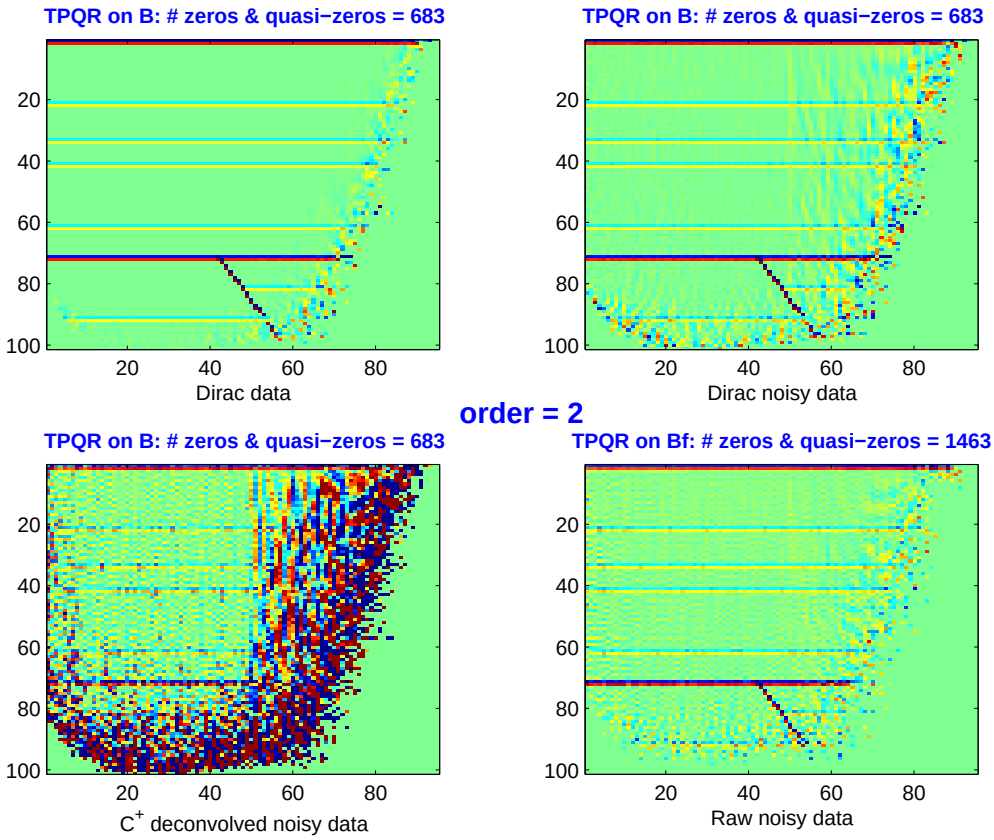


Figure 12: Optimal truncation ($\delta = 2$) for noisy data (subplots on the right). No progress for *a priori* deconvolved data (bottom left-hand subplot).

increasingly expensive to handle, in our case, this threshold could allow to stop at 65% of the total CPU time.

The optimal order of truncation, found to be 10^{-2} in our test case, is related to the 5% noise added to the data. Indeed, other experiments with a noise level of 1% lead to a slightly higher optimal truncation order ($10^{-2.5}$). Hence, with real data, the relative noise level must be estimated in order to set up a truncation threshold with the same order of magnitude.

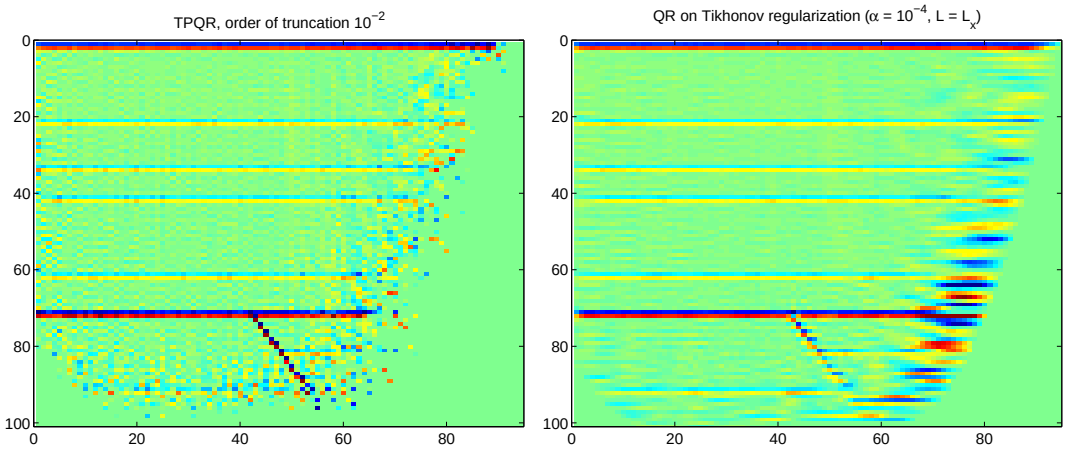


Figure 13: reflectivity obtained on noisy data (5% noise) with B_f.
left: by TPQR with truncation at 10^{-2}.
right: by sparse QR only, but on a problem regularized by Tikhonov. The regularizing term is $10^{-4}.\mathcal{L}_x$, the Laplacian operator in x which favours horizontal reflectors.

4.8 Sparse QR solution on a Tikhonov regularization

The previous results show the ability to handle migration with a truncated direct method, even when a convolution operator is inserted. However, this factorization is costly, namely because of the column pivoting. Alternatively, we could think of using the Tikhonov regularization [14] that will form a full-rank approximate operator. Then, straightforward QR factorization without pivoting becomes possible, allowing the use of the most efficient sparse techniques.

In this case, the objective function is changed by using a regularizing matrix L, $L \in \mathbb{R}^{p,m}$ and a positive factor α.

$$\begin{aligned} \min_r J(r) &= \| B \cdot r - d \|_2^2, \\ \text{is replaced by:} \quad \min_r \mathcal{J}(r) &= \| B \cdot r - d \|_2^2 + \alpha^2 \| L \cdot r \|_2^2. \end{aligned} \tag{20}$$

Definition (20) can easily be reshaped by setting up the following augmented least squares problem:

$$\begin{aligned} \min_r \mathcal{J}(r) &= \| \tilde{B} \cdot r - \tilde{d} \|_2^2, \\ \text{where:} \quad \tilde{B} &= \underbrace{\begin{pmatrix} \alpha L \\ B \end{pmatrix}}_{m} \begin{matrix} \}_p \\ \}_n \end{matrix} \quad \text{and } \tilde{d} = \begin{pmatrix} 0 \\ d \end{pmatrix} \begin{matrix} \}_p \\ \}_n \end{matrix} \end{aligned}$$

Depending whether the rank of L is equal or inferior to the dimension of the reflectivity space (m, size of r), a norm or a semi-norm is added to the initial cost function.

Choosing L as the identity matrix of $\mathbb{R}^m$ tends to minimize the norm of the solution. However, a very wide choice of regularizing functions is proposed in the literature. In our case, since most seismic profiles are recorded on horizontally stratified medium, the Laplace operator restricted to the horizontal direction (the partial second derivative in x) was selected. The simpler first derivative operator was also tested, but has proven less efficiency.

The horizontal Laplace operator is discretized by a 3-point template, that connects vertices of each horizontal line of the reflectivity grid,

$$\underbrace{\mathcal{L}_x}_{(n_x \cdot n_z, n_x \cdot n_z) \text{ matrix}} = \underbrace{\begin{pmatrix} 2 & -1 & 0 & \cdots & 0 \\ -1 & 2 & -1 & & \vdots \\ 0 & & \ddots & & \\ \vdots & & \ddots & \ddots & \\ 0 & \cdots & & & \ddots \end{pmatrix}}_{(n_x, n_x) \text{ matrix}} \otimes I_{n_z} \,. \tag{21}$$

When L is the identity matrix, the benefit of such a regularization lies in the fact that the smallest singular values of matrix $\tilde{B}$ are shifted. Indeed (same notations as equation (10)):

$$\forall i \in \mathbb{R}^n,\ \sigma_i(\tilde{B}) = \sqrt{(\sigma_i(B))^2 + \alpha^2}\,.$$

For L different from identity, the generalized singular value decomposition [13] becomes the relevant tool for the analysis, but the damping effect can also be evaluated.

Thus, the crux of this method lies in the choice of the adequate α, which can be related to the selection of the truncation order. Since the smallest singular values are now bounded downwards by α, and considering $\alpha \ll \sigma_1(B)$, then $\sigma_1(\tilde{B}) \approx \sigma_1(B)$. Therefore, to be consistent with the previous truncation threshold, we can select:

$$\alpha \approx 10^{-\delta}\sigma_1\,.$$

Then all the smallest singular values will be set to α.

From Figure 8, $\sigma_1(B)$ can be estimated at 10^{-2} (square root of the first diagonal element of the R factor in the PQR factorisation of the normal equation of B_f). σ_1 is also the 2-norm of B_f, for which many estimates are available (the maximum 2-norm of the columns of B_f, among others). Anyhow, this estimated value leads in our case to $\alpha = 10^{-4}$. Figure 13 displays the results in comparison with TPQR. The blurred area on the fringe of the domain is then covered by a coherent noise, due to the preferential horizontal direction, but on the other areas of the domain, the two images are very similar.

Complementary experiments are not displayed herein. With a smaller α, the results can also be compared to those obtained by TPQR with a lower truncation threshold: they induce a larger propagated data error, whereas a too large α blurs and wipes the image.

Although the present work goes beyond the search for a proper sparse linear least squares algorithm, a form of Tikhonov regularization has already been proposed for the migration operator in a fundamental paper on waveform inversion, [23]. There, the cost function is designed in a statistical framework involving two types of covariance matrices:

$$\jmath(r) = \frac{1}{2}(B \cdot r - d)^{\mathrm{T}} \cdot C_d^{-1} \cdot (B \cdot r - d) + \frac{1}{2} r^{\mathrm{T}} \cdot C_r^{-1} \cdot r \,, \text{with:}$$

$$C_d\left(d(R_i, S_j, t_k) | d(R_{i'}, S_{j'}, t_{k'})\right) = \sigma_{ijk}^2 \delta_{ii'} \delta_{jj'} \delta_{kk'} \,,$$

$$C_r\left(r(x,z) | r(x',z')\right) = \sigma_r^2 \exp\left(-\frac{1}{2}\left(\frac{(x-x')^2}{L_x^2} + \frac{(z-z')^2}{L_z^2}\right)\right).$$

C_d is a covariance matrix defined by the estimated errors on the data, considering the noise as uncorrelated. In the cost function, this diagonal matrix introduces a weight in the norm, with the possibility of setting up local confidence in the data for receiver R_i, at shot S_j and time sample t_k.

C_r describes some *a priori* knowledge on the statistical distribution of r, e.g. the anisotropy of the continuous reflectors. If L_x is set to Δx and L_z to a very small fraction of Δ_z, then C_r^{-1}, although having a dense structure, is very close to $\mathcal{L}_x$, the aforementioned horizontal Laplacian operator.

4.9 Lessons from the use of direct methods and application to the global inverse problem

From this study on direct factorization methods, it can be pointed out that an efficient sparse pivoted QR algorithm would be very handy, for solving the migration problem with a problem size at hand from nowadays computer power. While performing truncation out of the resulting factor, a satisfying answer can be given, even in cases propagation and convolution are combined.

The deconvolution issue has also been addressed carefully: the least squares solution of the combined operator cannot be properly found if deconvolution is performed *ab initio*. This implies a very expensive management of the fill-in for all direct methods, with respect whether storage or redundant computations. However, TPQR-QGS offers the most appropriate procedure among truncation methods.

As far as the global inverse problem in seismics is concerned, the aforementioned reduced least squares formulation (Equation (8), Section 2.2) can be managed by local optimization methods as long as it remains possible to compute the gradient of the objective function with respect to the background velocity. This involves the derivative of the pseudo inverse wrt. c, that truncation methods cannot offer. Alternatively, the Tikhonov regularization, with fixed damping operator and parameter, leads to the computation of a proper Moore-Penrose pseudo-inverse. In this case, the matrix has full rank and the straightforward

application of the normal equation induces a simple form for the derivative of the pseudo-inverse :

$$B(c)^\dagger = \left(B(c)^{\mathrm{T}} B(c)\right)^{-1} B(c)^{\mathrm{T}} ,$$
$$\Rightarrow \frac{dB^\dagger}{dc} = -B^\dagger \frac{dB}{dc} B^\dagger + \left(B^{\mathrm{T}} B\right)^{-1} \frac{dB}{dc}^{\mathrm{T}} \left(I - BB^\dagger\right) . \qquad (22)$$

The last term involves $P_{\mathcal{N}(B^{\mathrm{T}})} = I - BB^\dagger$, the orthogonal projection on the kernel of B^{T}. Since B is rectangular and vertical, although full rank, this projection is not the null operator.

From the computational point of view, equation (22) is not more awkward than dealing with the derivation of an inverse, because only two expensive steps arise: namely that the product with the pseudo inverse matrix has to be computed with two different vectors. In our framework, where the pseudo-inverse is defined by a Q-less QR factorization, we obtain the following algorithm (see below). Given a data set d and an increment vector of the background velocity δc, the gradient of the cost function can be applied by: (the frames highlight the most expensive computational steps),

$$\prec \nabla\mathbb{J}(c), \delta c \succ = < \left(B \cdot B^\dagger - I\right) d \ , \ \left((\frac{dB}{dc} \cdot \delta c) \cdot B^\dagger + B \cdot (\frac{dB^\dagger}{dc} \cdot \delta c) \right) d >$$

due to Q-less formulation:
$$\begin{cases} \left(B^{\mathrm{T}} B\right)^{-1} = R^{-1} \cdot R^{-\mathrm{T}} \\ B^\dagger = R^{-1} \cdot R^{-\mathrm{T}} \cdot B^{\mathrm{T}} \end{cases}$$

hence, compute first:
$$\begin{cases} r = B^\dagger \cdot d = \boxed{R^{-1} \cdot R^{-\mathrm{T}}} \cdot B^{\mathrm{T}} \cdot d \\ e = \left(B \cdot B^\dagger - I\right) d = B \cdot r - d \\ f = \left(\frac{dB}{dc} \cdot \delta c\right) \cdot r \end{cases}$$

then: $g = (\frac{dB^\dagger}{dc} \cdot \delta c) \cdot d = \boxed{R^{-1} \cdot R^{-\mathrm{T}}} \left(-B^{\mathrm{T}} \cdot f - (\frac{dB}{dc} \cdot \delta c)^{\mathrm{T}} \cdot e \right)$

$$\Rightarrow \prec \nabla\mathbb{J}(c), \delta c \succ = < e \ , \ f + B \cdot g > \ .$$

This perspective advocates the use of the Tikhonov regularization, together with the availability of very efficient sparse QR solvers without pivoting. However, specific implementation is still needed to incorporate a convolution on the fly.

5 Practical solutions by iterative solvers

The size of the linear least squares problem generated by the pre-stack depth migration obviously favours the use of iterative solvers. These algorithms have already been quoted and implemented in several research works, [23], [21], [22], [17]. In most cases, a maximum number of iterations constitutes the stopping criterion rather than any threshold on the convergence: the considered maximum usually remains very low ($\simeq 10$).

The extreme case consists in performing only one iteration of a well-preconditioned gradient: this procedure has been shown to be equivalent to a weighted form of the Kirchhoff migration, the algorithm of choice among those implemented in commercial geophysical software [23], [24].

In this work, however, the aim consists in exhibiting an accurate solution, accepting the price of the required number of iterations in order to achieve a good convergence. Thanks to the experience gained in the previous section about this discrete ill-posed problem, the study described thereafter is really focused on the iterative process. Indeed:

- deconvolution is never performed *ab initio*: this strategy adopted in stability issues has a low cost, since the convolution operator can be inserted on the fly in iterative solvers of the conjugate gradient type, despite the need for a specific implementation, (further notation B will stand for $B_f = C_f \cdot B$).
- a regularizing procedure is compulsory with noisy data: we can either apply the Thikonov regularization as is, or utilize the counterpart of the truncation for direct methods, i.e. the limitation of the number of iterations. Some recent works will be quoted, as they propose heuristic error estimates to determine the optimal iteration count.

This section is devoted to the comparison of several Conjugate Gradients like algorithms regarding various concerns: convergence rates, error estimates, regularization, preconditioners, adequation of the procedure to fit into the global non-linear inverse problem.

5.1 Conjugate gradient algorithms for least squares

The following algorithms have in common that they only involve two matrix-vector products per iteration, one with B, another with B^{T}. The time spent on these two operations accounts for most of the computational cost of an iteration: we recall below the total number of BLAS 1 operations (scalar products and vector linear combinations) together with the length of the vectors involved. The storage issue of intermediate vectors is even more questionable and will thus be highlighted.

Note that in the sequel, our customary unknown, namely the reflectivity r, will be named x, in order to preserve the usual denomination for the residual of iterative methods.

Moreover, let us recall the minimization properties of the initial conjugate gradient algorithm acting on a symmetric positive definite matrix A.

$$\begin{aligned}
&\text{let } \hat{x} \text{ be the solution of: } A\hat{x} = b \ , \\
&\mathsf{CG} \text{ minimizes the } A\text{-norm of the error, i.e., the quantity:} \\
&\epsilon_k = (x_k - \hat{x})^{\mathrm{T}} A (x_k - \hat{x}) \ , \\
&\text{and equivalently the } A^{-1}\text{-norm of the residual:} \\
&\rho_k = (b - Ax_k)^{\mathrm{T}} A^{-1} (b - Ax_k) \ .
\end{aligned} \tag{23}$$

For the least squares problem, there are two main formulations which, in euclidian norm, minimize either the residual or the true error. In all cases, the normalized quantity

$$\mu_k = \frac{\| Bx_k - d \|_2}{\| Bx_0 - d \|_2} , \tag{24}$$

is not going to vanish to zero, since the data d do not generally belong to the range of B. However, the following normalized quantity, issued from the normal equations, can potentially converge to zero:

$$\nu_k = \frac{\| B^{\mathrm{T}}(Bx - d) \|_2}{\| B^{\mathrm{T}}(Bx_0 - d) \|_2} . \tag{25}$$

Readers familiar with high performing numerical solvers may be disappointed by the results displayed with normalized ratio (25) in Figure 14. Namely, the computations performed in double precision arithmetic are deemed to be satisfactory for a convergence reaching 10^{-5} in relative value at the most. The need for regularization will justify this low demand.

Only the most convincing algorithms will be displayed below, with the appropriate references. Some specific steps have been inserted, among which the computation of heuristic error estimates that will be discussed later on. The stopping criterion is based in any case on a maximum number of iterations and a targeted convergence of the ratio (25).

- CGLS: the Conjugate Gradient Least Squares method, [2], corresponds to applying the regular conjugate gradient to normal equations, therefore also called CGNE in [12]:

  $$\text{solve: } B^{\mathrm{T}} \cdot Bx \;=\; B^{\mathrm{T}} d \, .$$

  This method minimizes the residual error on the Krylov subspaces built with $B^{\mathrm{T}}B$, since, from (23):

  $$\begin{aligned} \epsilon_k &= (x_k - \hat{x})^{\mathrm{T}} B^{\mathrm{T}} B (x_k - \hat{x}) \, , \\ &= (Bx_k - d)^{\mathrm{T}} (Bx_k - d) \;=\; \| Bx_k - d \|_2 \, . \end{aligned}$$

  Therefore, CGLS is also called CGNR for Conjugate Gradient Normal Residual.

  Remembering that $m \gg n$, in addition to the multiplications by B and B^{T}, there are respectively 2 and 3 BLAS1 operations per iteration on vectors of length n and m, and 2 and 3 intermediate vectors of respective length are to be used.

  LSQR is another algorithm known to be more robust than CGLS, which follows in perfect arithmetic the same convergence path. LSQR, [15], is a Lanczos version of CGLS, which is based on a bidiagonalization of matrix B. No noticeable improvement occurs when tested in this case.

- ORTHODIR-CR: as it is called in [5] or MR-II as it is referred in [12] is an algorithm belonging to the family of minimum residual algorithms together with CGLS. ORTHODIR-CR is characterized by a two term recurrence, that allows to handle symmetric indefinite matrices.

Input: a matrix B of dimensions (m, n), a relative stopping criterion $\bar{\varepsilon}$, a maximum number of iterations $k_{\max}$, a right-hand side d, an initial guess x_0.

1: $r_0 = d - Bx_0$ {initial residual}
2: $p_0 = B^{\mathrm{T}} r_0$ {first descent direction}
3: $\nu_0 = \| p_0 \|_2$; $\varepsilon = \bar{\varepsilon}\nu_0$ {normalised stopping criterion}
4: $q_0 = Bp_0$ {first residual direction}
5: $\alpha_0 = \| q_0 \|_2^2 / \nu_0^2$ {step size}
6: $\pi_0 = 0$; $\pi_1 = \alpha_0$ {$|\Pi_k'(0)|$ for error estimate}
7: **while** $\nu_k \geq \varepsilon$ and $0 \leq k \leq k_{\max}$ **do**
8: $\quad x_{k+1} = x_k + \alpha_k p_k$ {solution update}
9: $\quad r_{k+1} = r_k - \alpha_k q_k$ {residual update}
10: $\quad t_{k+1} = B^{\mathrm{T}} r_{k+1}$ {projected residual update}
11: $\quad \nu_{k+1} = \| t_{k+1} \|_2$ {projected residual}
12: $\quad \xi_{k+1} = \sqrt{\pi_{k+1}}\, \nu_{k+1}$ {error estimate}
13: $\quad \beta_{k+1} = \nu_{k+1}^2 / \nu_k^2$
14: $\quad p_{k+1} = t_{k+1} + \beta_{k+1} p_k$ {descent direction}
15: $\quad q_{k+1} = Bp_{k+1}$ {residual direction}
16: $\quad \alpha_{k+1} = \| q_{k+1} \|_2^2 / \nu_{k+1}^2$ {step size}
17: $\quad \pi_{k+2} = \pi_{k+1} + \alpha_{k+1} (1 - \beta_{k+1}/\alpha_k) (\pi_k - \pi_{k+1})$ {$|\Pi_k'(0)|$ update}
18: $\quad k \hookleftarrow k + 1$
19: **end while**

Output: k, x_k, ν_k and ξ_k.

Algorithm 2: CGLS (CGNE, CGNR) algorithm for least squares problem.

Algorithm 3 straightforwardly replaces the symmetric matrix by the normal equation matrices, although it could have been rewritten while keeping the usual residual $r = Bx - d$. However, as for CGLS, the stopping criterion is based upon the residual of the normal equations (25), and not on the true residual (24). Hence, both matrix-vector products are performed at step 16 on the same variable.

In the original algorithm, the first descent direction (step 4) can either be $p = B^{\mathrm{T}} B r_0$ or $p = r_0$. In [5], the second alternative is preferred from numerical experiments, with the restriction that this first vector should preferably belong to the range of the iteration matrix. Step 1 indicates here that r_0 belongs to the range of B^{T}. Since the assertion $\mathcal{R}(B^{\mathrm{T}}) = \mathcal{R}(B^{\mathrm{T}} B)$ always holds, the choice $p = r_0$ complies with the aforementioned advice. Our numerical experiments have confirmed the relevance of this strategy. There are respectively 5 and 1 intermediate vectors of respective length n and m to be used. All 9 BLAS1 operations per iteration are performed on vectors of length n. Somehow, the same economy can be made if CGLS is written as a blank CG on the normal equations.

Input: a matrix B of dimensions (m, n), a relative stopping criterion $\bar{\varepsilon}$, a maximum number of iterations $k_{\max}$, a right-hand side d, an initial guess x_0.

$r_0 = B^{\mathrm{T}}(d - Bx_0)$ {initial residual}
$\nu_0 = \| r_0 \|_2$ {normalised stopping criterion}
$\varepsilon = \bar{\varepsilon}\nu_0$ {normalised stopping criterion}
$p_{-1} = 0\,;\ p_0 = r_0$ {first descent directions}
$q_{-1} = 0\,;\ q_0 = B^{\mathrm{T}}Bp_0$ {first residual directions}
$\eta_0 = \| q_0 \|_2$
$\pi_0 = \omega_0 = 0\,;\ \varpi_1 = 1./\eta_0$ {$\Pi_k''(0)$ for error estimate}
while $\nu_k \geq \varepsilon$ and $0 \leq k \leq k_{\max}$ **do**
  $\alpha_k = \left(r_k{}^{\mathrm{T}} \cdot q_k\right) / \eta_k{}^2$ {step size}
  $x_{k+1} = x_k + \alpha_k p_k$ {solution update}
  $r_{k+1} = r_k - \alpha_k q_k$ {residual update}
  $\nu_{k+1} = \| r_{k+1} \|_2$ {residual norm}
  $\pi_{k+1} = \pi_k - 2\alpha_k\eta_k\omega_k$ {$\Pi_k''(0)$ for error estimate}
  $\chi_{k+1} = |\pi_{k+1}|\ \nu_{k+1}$ {error estimate}
  $\sigma_k = \eta_k/\eta_{k-1}$
  $t_k = B^{\mathrm{T}}Bq_k$
  $\gamma_k = \left(q_k{}^{\mathrm{T}}t_k\right) / \eta_k{}^2$
  $p_{k+1} = (1./\eta_k)\, q_k - (\gamma_k/\eta_k)\, p_k - \sigma_k p_{k-1}$ {descent and residual directions}
  $q_{k+1} = (1./\eta_k)\, t_k - (\gamma_k/\eta_k)\, q_k - \sigma_k q_{k-1}$ {2 step recurrence update}
  $\eta_{k+1} = \| q_{k+1} \|_2$
  $\omega_{k+1} = \varpi_{k+1}/\eta_{k+1}\,;\ \varpi_{k+2} = -\gamma_k\omega_{k+1} - \eta_{k+1}\omega_k$ {for error estimate}
  $k \hookleftarrow k + 1$
end while

Output: k, x_k, ν_k and χ_k.

Algorithm 3: ORTHODIR-CR algorithm for least squares problem.

- CGME: in a dual approach to CGLS, the conjugate gradient with minimum error expresses the following two step algorithm:

$$\begin{cases} \text{solve by CG: } BB^{\mathrm{T}}w = d\,, \\ \text{then, set: } x = B^{\mathrm{T}}w\,. \end{cases}$$

  BB^{T} is a positive semi-definite matrix, with a huge kernel in our case. It can however be very attractive if the data belongs to the range of B, because it minimizes the true euclidian error (hence the denomination):

$$\text{if: } B\hat{x} = d \text{ and } \hat{x} = B^{\mathrm{T}}\hat{w}\,, \text{ then CGME minimizes:}$$

$$\| w - \hat{w} \|_{BB^{\mathrm{T}}} = (w - \hat{w})^{\mathrm{T}}BB^{\mathrm{T}}(w - \hat{w}) = \| x - \hat{x} \|_2\,.$$

  The actual algorithm is not reproduced herein, since somewhat expectedly the results for our problem are not encouraging, due to the fact that the data does not belong

to the range of the operator. However, the relation between CGME and the next algorithm justifies this short presentation.

The number of BLAS1 operations and of vector storages is reciprocal to the number in CGLS, with respect to the vector lengths m and n, which is bad news since $m \gg n$.

- ORTHODIR-CR+MBTT: aimed at providing a framework for the solution of the global inverse problem, the Migration Based Travel formulation of the cost function has been explained in section 2.2, and equation (7) leads to:

$$\begin{cases} \text{find:} & \min_w \| BK^{-1}B^{\mathrm{T}}w - d \|_2 \ , \\ \text{then, set:} & x = B^{\mathrm{T}}w \ , \end{cases}$$

where K is a positive diagonal matrix approximating $B^{\mathrm{T}}B$. A conjugate gradient algorithm with iteration matrix $A = BK^{-1}B^{\mathrm{T}}$ then becomes a preconditioned version of CGME. The knowledge that $B^{\mathrm{T}}B$ cannot be full rank has already led us to use the ORTHODIR-CR algorithm on this symmetric indefinite operator in previous work.

A diagonal matrix K of the following kind has been selected:

$$K = diag(B^{\mathrm{T}}B) + \alpha I_m \ ,$$

which can be inverted for any positive α.

$\| BK^{-1}B^{\mathrm{T}}s - d \|_2$ is the residual displayed for ORTHODIR-CR+MBTT in Figure 14. Its definition therefore differs somewhat from the other ones.

All the BLAS1 operations and storage are concerned with vectors of length m. In section 5.5, it will be shown that the computation of time reflectivity w (which motivated the use of this algorithm) can be obtained by a slight modification of CGLS or ORTHODIR-CR.

Figure 14 offers two complementary ways of considering the results. On the left-hand side, the evolution of the normalized residual is plotted in logarithmic scale, while the true error on the resulting reflectivity is displayed on the right-hand side.

In practice, only the residual should be at our disposal. ORTHODIR-CR shows then the most convincing behaviour: the residual decreases on a very regular curve, and reaches the lowest values. CGLS also performs relatively well: although the path is slightly irregular, with spikes of constant intensity in logarithmic scale, there seems to be no difficulty in deciding a confident stopping criterion for this algorithm. On the contrary, the residuals of CGME and ORTHODIR-CR+MBTT are stalled after 50 iterations.

Since the true error is at hand for these numerical experiments, the previous conclusions are somehow overturned. First, all error curves are U-shaped and very regular. Thus, an optimal iteration count exists for each of these algorithms, reached at circa 200 iterations for a problem size of some 8000 unknowns: the coordinates of these optimal points are indicated on the plot (iteration number, distance to the true solution). In fact, too many

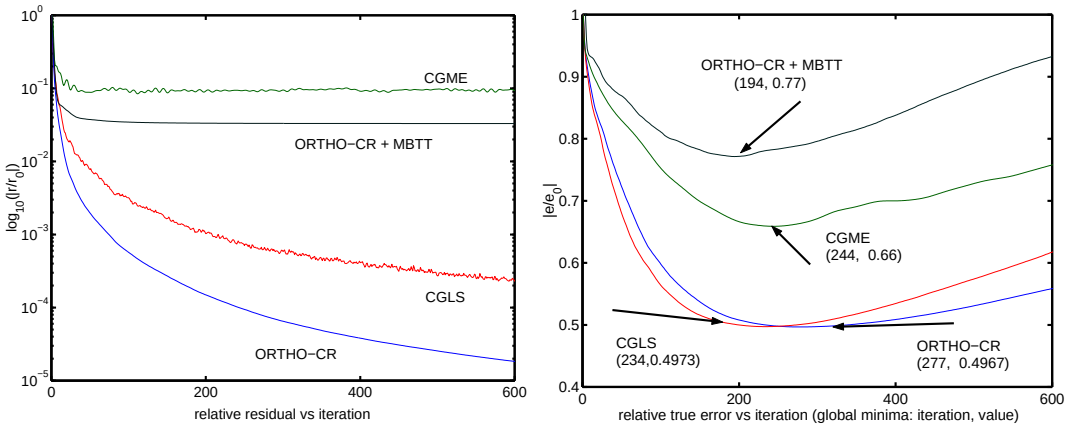


Figure 14: Comparison of conjugate gradient algorithms for the migration problem. The direct operator includes the convolution and the data has a 5% noise. Tested algorithms: CGLS(LSQR), ORTHO-CR, CGME and ORTHO-CR+MBTT.
left: Convergence of relative residual in logarithmic scale.
right: Convergence of relative true error in linear scale.

iterations bring back the perturbations that have been observed with the direct methods without regularization.

With respect to this relative true error, admitting that there is an adequate stopping criterion which retains the optimal iteration count, then CGLS and ORTHODIR-CR achieve the best performances. CGLS could be preferred for a faster convergence, whereas ORTHODIR-CR cannot benefit from a slightly better vicinity to the true solution. Behind, CGME gets closer than ORTHODIR-CR+MBTT, but the main observation is that in both cases, the minimal error is attained long after the residual is stalled.

5.2 Regularisation by limited convergence

As for the truncation threshold or for the parameter in the Tikhonov damping factor, the noise level in the data (or the distance between the data and the range of the direct operator) determines the optimal number of iterations, [12]. With CGLS and 5% noise in the data, Figure 15 compares the reflectivity fields obtained after 1000 and 250 iterations, the latter value deriving from an error estimate. The regularizing effect of an early stop is obvious. The optimal iteration count with respect to the true error would have been 234 in this case, but the difference could not be noticed on this image (less than 0.5% difference over the true error!).

In [12], an error estimate is proposed, which is based on the so-called *residual polynomials* $\{\Pi_k\}$ that express the linear relation between the initial residual and the residual of the k^{th}

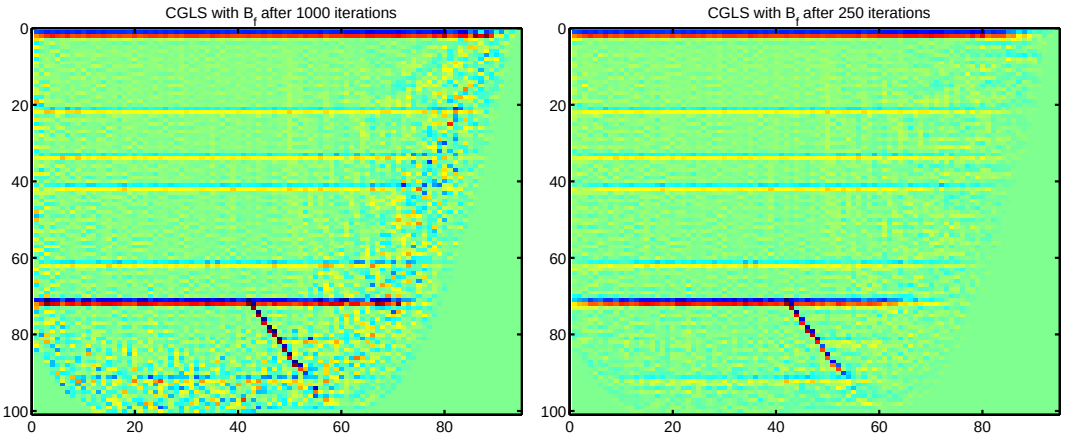


Figure 15: Resulting reflectivity after too many iterations (left), or stopped according to an error estimate (right).

iteration of a Krylov method :

$$b - Ax_k = \Pi_k(A)(b - Ax_0) \ .$$

Each of these polynomials has the corresponding degree k. Moreover, if for the Krylov method, one defines the iterates by:

$$\begin{aligned} x_{k+1} &= x_k + \alpha_k p_k \ , & \{p_k\} \in \mathbb{R}^n, \ \{\alpha_k\} \in \mathbb{R}, \\ p_{k+1} &= b - Ax_{k+1} + \beta_{k+1} p_k \ , & \{\beta_k\} \in \mathbb{R}, \end{aligned}$$

then the residual polynomials follow a two term recurrence:

$$\Pi_{k+1}(X) = \Pi_k(X) - \alpha_k X \cdot \Pi_k(X) - \alpha_k \frac{\beta_k}{\alpha_{k-1}} \left(\Pi_{k-1}(X) - \Pi_k(X)\right) \ . \tag{26}$$

On the basis of the optimal properties of these polynomials, heuristic error estimates are derived:

- for CGLS: $\xi_k = \sqrt{|\Pi'_k(0)|} \ \| \ B^{\mathrm{T}}(Bx_k - d) \ \|_2$. In [12], the original estimate takes the actual residual $\| \ Bx_k - d \ \|_2$, but this value is much too rapidly bounded to a limit downwards while $\sqrt{|\Pi'_k(0)|}$ is a growing factor. The product of both expressions gives a steadily increasing value. Thus we thought of substituting the projected residual.
- for ORTHODIR-CR: $\chi_k = |\Pi''_k(0)| \ \| \ B^{\mathrm{T}}(Bx_k - d) \ \|_2$. The previous estimate ξ_k cannot be defined for this algorithm, since it can be shown that $|\Pi'_k(0)| = 0$ for all k, therefore the higher order of derivation.

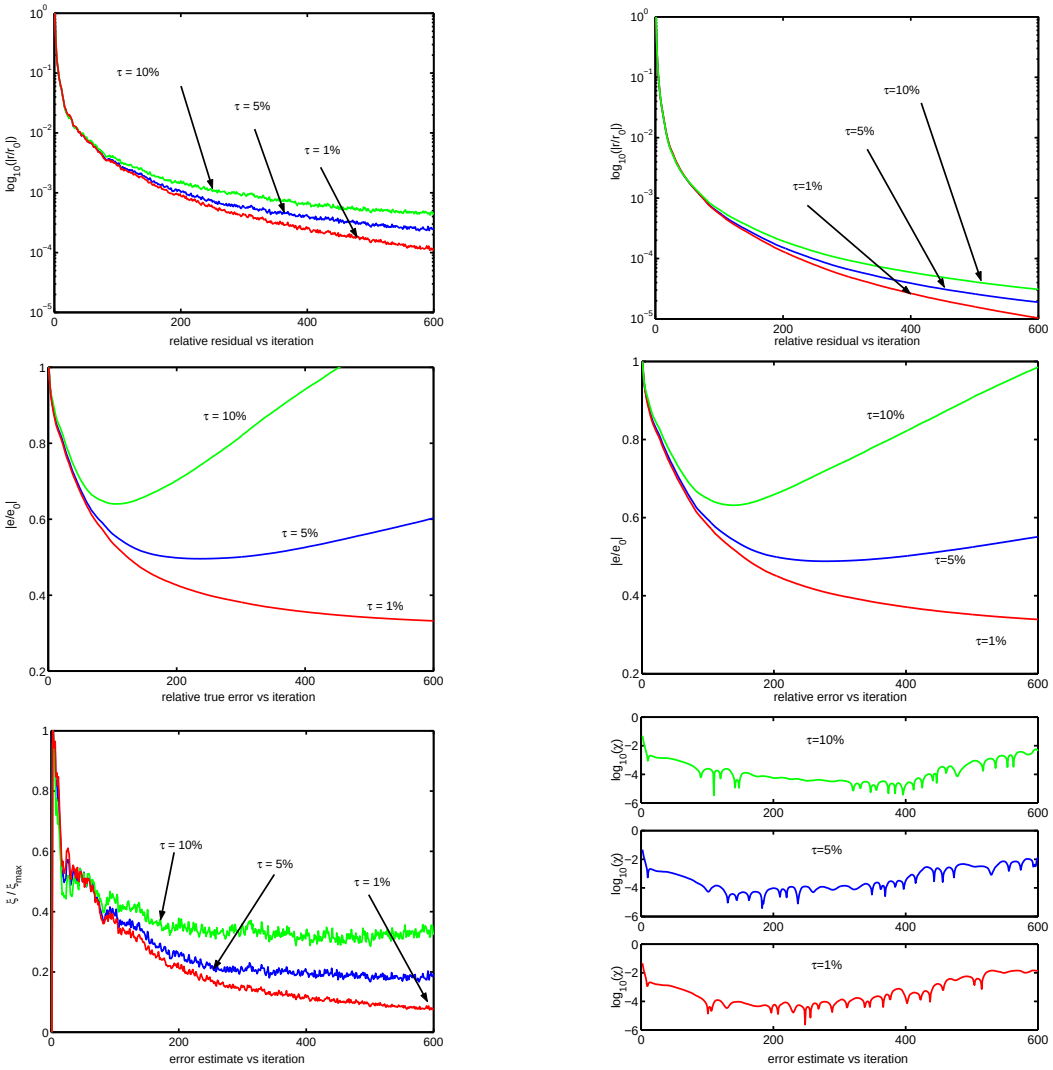


Figure 16: Performance of error estimates with varying noise levels.
left: on CGLS, *right*: on ORTHODIR-CR.
top: residual, *centre*: true error, *bottom*: error estimate.

These expressions can be evaluated at each step at a low computational cost, from scalar recurrence formulae calculated from (26). Steps 6, 12 and 17 of Algorithm 2, steps 7, 13, 14 and 21 of Algorithm 3 are devoted to these estimates: although these lines complicate the

reading, it was thought that it was worth displaying full practical algorithms for once. To clarify the notations, all vectors are Latin letters while scalars are Greek ones.

According to [12], in both cases the heuristic stopping rule should be to stop whenever these estimates (which should rather be called indicators) reach a global minimum. However, in practice, we have observed that the relevant information has to be based upon the following:

- for ξ_k: plot this value in linear scale then identify the end of the decreasing slope before the value is stalled,
- on χ_k: plot this value in log scale then identify the centre of the trough.

However, a significant validation is obtained by varying the noise level as reported on Figure 16, in a comparison between CGLS (left column) and ORTHODIR-CR (right column).

- on the top sketches, the normalized projected residuals are plotted. No stopping criterion can be extracted from the analysis of this parameter, and the smoother and faster convergence of ORTHODIR-CR does not guarantee a smaller error than for CGLS;
- the central sketches display the true errors, which are available only in the framework of such a validation process: as mentioned above, the larger the noise level, the earlier one should stop, but for 1% noise, the optimum is not reached after 600 iterations of any of the two algorithms, which somehow always behave similarly with respect to the true error.
- the bottom line shows the behavior of the error estimates. For CGLS (bottom left sketch), this information seems relevant. Indeed, the tips of the label arrows target the area where these spiky curves begin to level off: although not very acute, this criterion follows the trend of the true error with respect to the noise level. For ORTHODIR-CR however, the behaviour of the estimate is not easy to analyze: the three curves are plotted separately since they cross each other several times; moreover, the dependency seems to be the reverse of what was expected.

Thus, despite a faster residual convergence for ORTHODIR-CR than for CGLS, the latter algorithm shows a slightly better behaviour with respect to the true error but above all, CGLS disposes of a more efficient heuristic error estimate for stopping at a convenient regularization level.

Moreover, our problem might not require the design of a more reliable and more precise criterion for the optimal iteration count. Since between 150 and 300 iterations, the curve of the true error is very flat, any criterion designating a stop in this range is acceptable.

5.3 Additional Tikhonov regularization

Albeit, as just quoted, the regularization by limited convergence of iterative methods always occurs, a Tikhonov regularizing term can be added to the initial least squares function, especially when some knowledge about the solution distribution exists.

Using the same function as in Section 4.8 (namely, the Laplacian in x), the results with a range of coefficients are displayed on Figure 17. This additional regularizing tool does not enhance the performance of the optimal solution as much as expected. With respect to the true error (left-hand sketch, Fig. 17), a slightly better result is obtained when $\alpha = 10^{-4}$ at the price of a longer convergence path. This coefficient is identical to the one chosen with direct methods.

Furthermore, even though the error estimate (right-hand sketch, Fig. 17) still performs well in indicating where to stop or not, the various curves cannot be interpreted in order to decide which parameter has to be chosen in front of the Tikhonov term.

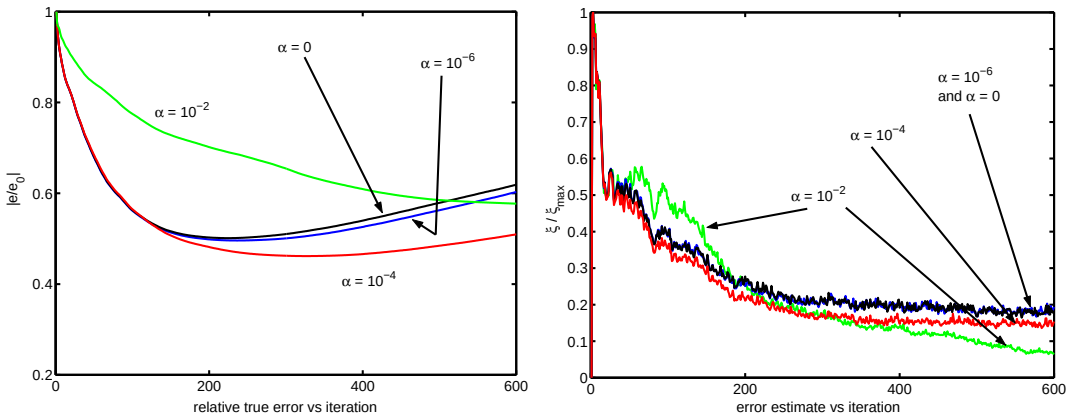


Figure 17: Parametric study of damping factors for Tikhonov regularization, with a noise level of 5%, and computation by CGLS.
left: evolution of the relative true error versus iteration,
right: behavior of the error estimate.

On the so-called *L-curves*, extensively described in [14], the identification of the optimal Tikhonov parameter can be performed thanks to a plot over the following coordinates:

$$\left(\log_{10}(\| Bx^{\alpha}_{\text{opt}} - d \|_2) \ , \ \log_{10}(\| \mathcal{L}_x \, x^{\alpha}_{\text{opt}} \|_2)\right) \ . \tag{27}$$

An L shaped parametric curve should link the values obtained for an increasing α, where the main elbow in the lower left corner locates the optimal value. This solution embodies a trade-off between smooth characteristics with respect to $\mathcal{L}_x$ and a low residual with the initial operator.

In the part of this study devoted to direct solvers, the CPU cost of one factorization was already so expensive that the exploration over a full set of Tikhonov parameters was out of reach. While it becomes feasible with iterative solvers, another difficulty arises: how many iterations are necessary in order to build a consistent solution? Figure 18 shows the results obtained after 250 iterations, for this number belongs to the range of convergence designated in the previous section.

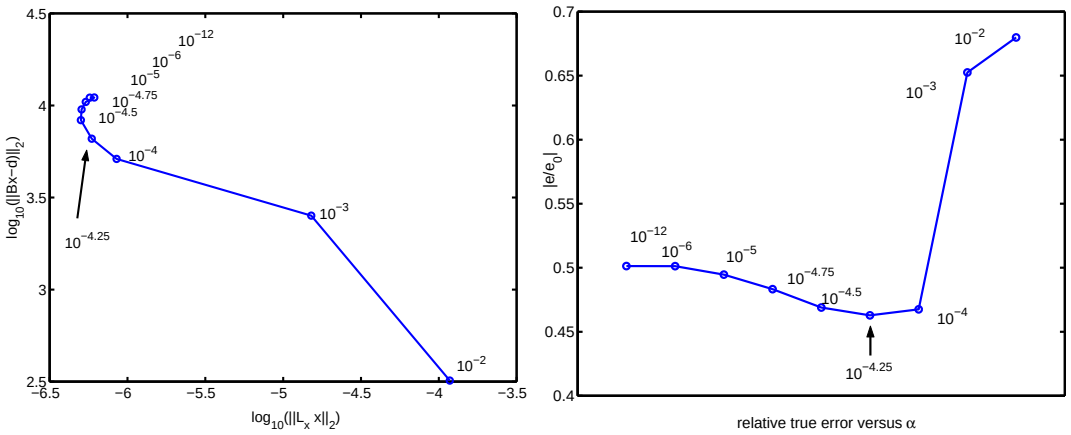


Figure 18: Search for the optimal Tikhonov parameter with CGLS on data with 5% noise: $\min_r \mathcal{J}(r) = \| B \cdot r - d \|_2^2 + \alpha^2 \| L \cdot r \|_2^2$.
left: L-curve drawn with coordinates (**??**), an elbow can be identified;
right: true error versus the Tikhonov parameter, the minimum of which corresponds to the elbow.

The parametric curve on the left of Figure 18 shows a vague L-shape. Still, the interval $[10^{-4.25}, 10^{-4}]$ can be selected for the optimal value of α, because it belongs to the elbow of the curve at the closest distance to the lower left corner. The fact that the residual is not a strictly growing function of α is due to the fixed number of iterations for solving problems of varying convergence rates: this is one of the difficulties in drawing an L-curve with results based on iterative methods.

The display of the true error (right sketch, Fig. 18), indicates that $\alpha = 10^{-4.25}$ is the optimal value, but once again, this information is available for validation purpose only.

In any case, it can be considered that the Tikhonov regularization on top of an iterative method does not contribute much to the solution of our problem. However, if its use is justified by some a priori knowledge of statistical properties of the solution, a heuristic based on a L-curve drawing is feasible.

5.4 Remarks about preconditioning

For all these iterative algorithms, let us recall that a matrix-free implementation can replace the use of the sparse storage which has been chosen for this study, problem size permitting. Nevertheless, the use of a proper matrix allows the construction of many types of preconditioners.

Our experience on that matter has revealed that none of the common preconditioning techniques can convincingly improve the convergence. Diagonal preconditioners, incomplete

Cholesky factor of the normal equations, incomplete R factor in the QR factorization have been tested in vain. Because of the huge problem size, the strategy for incompletion was based on geometry: that is to say, the coefficients are preserved whenever their indices correspond to neighbouring vertices on the reflectivity grid. The alternative consisting in evaluating the highest coefficients in the normal equation matrix has also failed.

Indeed, when looking at the convergence path on the relative residuals, the first iterations do not waste any time. Therefore, preconditioners which are mostly efficient to help in that matter do not perform well on our problem. However, if the operator does not include the convolution, then the preconditioners based on incomplete factors show some efficiency, but it has already been shown that the approach with initial deconvolution is not relevant.

Finally, one can also wonder whether the usage of preconditioners is not antinomic with the regularization by a limitation of the convergence. Indeed, with an efficient preconditioner on a symmetric matrix, the components of the solution on the smallest eigenvectors are displayed earlier in the convergence path, while they generate the unwanted instabilities.

5.5 Lessons from the use of iterative solvers and application to the global inverse problem

Iterative solvers remain the economical way for solving our very large linear least squares problem, all the faster as the iterations have to be stopped before the residual actually converges, because of regularization. The difficulty rests in designing a proper stopping criterion.

Several algorithms have been tested, together with a heuristic error estimate. By slightly modifying an estimate already proposed in the literature, we have found that CGLS is the easiest algorithm to tune, closely followed by ORTHODIR-CR.

Owing to this powerful inherent regularization property, on the one hand, the alternative by the Tikhonov regularization has a moderate interest, on the other hand, acceleration by preconditioning has not proven any efficiency and might even not be desirable.

With respect to the global inverse problem, a reduced least square approach subject to Equation (8), Section 2.2 can only be considered by means of the construction of an adjoint state if iterative methods are employed, as proposed in [22]. The number of iterations quoted in the previous sections and the vector length of the reflectivity forbid the use of such a technique. Conversely, the MBTT formulation, briefly described in section 2.2 was introduced in order to enlarge the domain of attraction of the global minimum of the complete inverse problem. This is performed by a *change of variable*, namely setting $x = B^{\mathrm{T}}w$ and keeping the vector w of much larger size ($n \gg m$) as the new linear unknown.

It has already been shown that ORTHODIR-CR on the MBTT matrix BKB^{T} did not converge well (see Fig. 14). Figure 19 is even more convincing, by showing the blurred optimal reflectivity obtained by ORTHODIR-CR + MBTT compared to the focused one deriving from the simple least squares and ORTHODIR-CR.

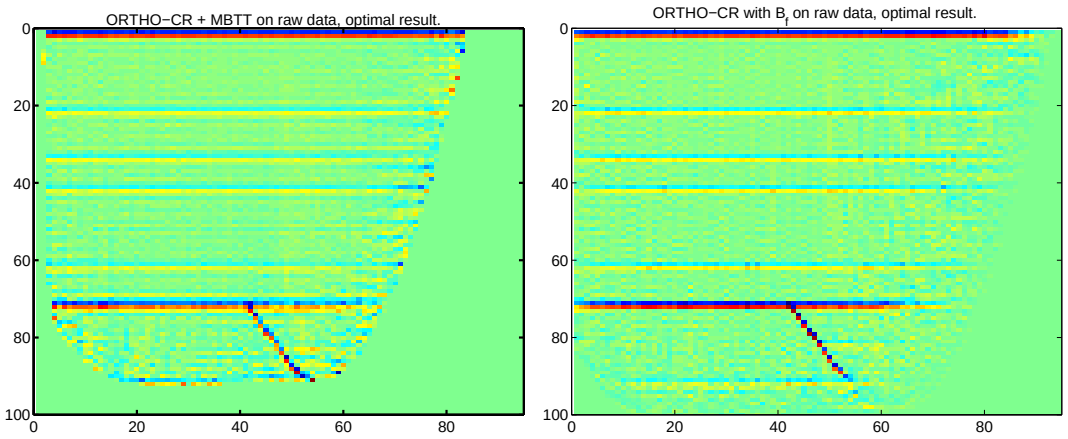


Figure 19: Optimal result with the MBTT formulation (left), and with the simple least squares formulation (right).

However, by means of two additional variables, it is possible to modify the CGLS algorithm in order to preserve the MBTT formulation. Indeed, on Algorithm 2, the following modifications should be applied or inserted:

- an initial guess w_0 of the time reflectivity w can be provided (d is a good choice) instead of a space reflectivity x_0 ;
- step 0' sets $x_0 = B^{\mathrm{T}} w_0$;
- step 1' sets $v_0 = r_0$;
- step 9' sets $w_{k+1} = w_k + \alpha_k v_k$ {MBTT solution update};
- step 14' sets $v_{k+1} = r_{k+1} + \beta_k v_k$;

Two additional vectors of length n have to be stored and updated by linear combination at each step. The final value of w can in turn be used as fixed for the alternate step of the MBTT algorithm, namely the non-linear optimization with respect to the background velocity. Consequently, a convenient answer can be proposed to the global inverse problem with an iterative solver for the linear least square.

6 Conclusion

In the framework of waveform inversion, it has been recalled that the migration of reflection seismic data can be set up as the solution of a linear least squares problem. An insight into the associated operator reveals a discrete ill-posed problem of a very large size, with a

noteworthy sparsity of the matrix. Thereafter, not only matrix-free procedures are practical, but also the whole toolbox of linear algebra for least squares.

With particular reference to the compulsory incorporation of the convolution operator, direct and iterative methods have been adapted in order to extract more than a customary approximate solution. The need for regularization becomes then a key issue for the adequacy of the result provided. Efficient and tunable algorithms have then been selected in both families of numerical solvers, namely TPQR as a direct method and CGLS as an iterative one.

However, the choice of these two most appropriate algorithms can be reconsidered if the migration is embedded in a global inversion, for instance when the background velocity has to be identified as well as the reflectivity. Indeed, whenever local optimization methods are chosen as non-linear solvers, either the differentiation of the solution of the linear least squares must remain feasible, or the MBTT formulation can be employed. These considerations have designated any efficient QR sparse factorization on a damped Tikhonov regularized problem as a direct solver and, as regards iterative methods, a version of CGLS that preserves a dual variable in the data space.

These extensive numerical experiments have been made feasible on a subset of the actual problem, which is however significant in size as far as common studies on least squares solvers are concerned. A balanced validation with a CPU efficient implementation on real data is the natural follow-up to this study, leaving the challenge open between direct and iterative techniques.

References

[1] P. R. Amestoy, I. S. Duff, and C. Puglisi. Multifrontal QR factorization in a multiprocessor environment. *Numerical linear algebra with applications*, 3(4):275–300, jul /aug 1996.

[2] A. Björck. *Numerical methods for least squares problems.* SIAM, 1996.

[3] N. Bleistein. *Mathematical methods for wave phenomena.* Academic Press, New York, 1984.

[4] G. Böhm and A. L. Vesnaver. In quest of the grid. *Geophysics*, 64(4):1116–1125, july-aug. 1999.

[5] D. Calvetti, L. Reichel, and Q. Zhang. Conjugate gradient algorithms for symmetric inconsistent linear systems. In J.D. Brown, M.T. Chu, D.C. Ellison, and R.J. Plemons, editors, *Cornelius Lanczos International centenary conference*, pages 267–272, Philadelphia, 1994. SIAM.

[6] T.F. Chan. Rank revealing QR factorizations. *Linear Algebra Applications*, 88/89:67–82, 1987.

[7] G. Chavent. Duality method for waveform inversion. Technical Report 2975, INRIA, Le Chesnay, France, 1996.

[8] G. Chavent, F. Clément, and S. Gómez. Waveform inversion by migration based travel time formulation. In *3rd Int. Conf. on mathematical and numerical aspects of wave propagation*, pages 1179–1182, Philadelphia, 1995. SIAM.

[9] J.F. Claerbout. Towards a unified theory of reflector mapping. *Geophysics*, 36:467–481, 1971.

[10] I.S. Duff, A.M. Erisman, and J.K. Reid. *Direct methods for sparse matrices*. Monographs on numerical analysis. Clarendon press, Oxford, 1986.

[11] G. Golub and C. Van Loan. *Matrix computations*. North Oxford Academic, 1983.

[12] M. Hanke. *Conjugate gradient type methods for ill-posed problems*. Pitman Research Notes in Mathematics. Longman Scientific and Technical, Harlow, 1995.

[13] P. C. Hansen. Regularization, gsvd and truncated gsvd. *BIT*, 29:491–504, 1989.

[14] P. C. Hansen. Analysis of discrete ill-posed problems by means of the l-curve. *SIAM Review*, 34(4):561–580, December 1992.

[15] C. C. Paige and M. A. Saunders. LSQR: an algorithm for sparse linear equations and sparse least squares. *ACM Transactions on Mathematical Software*, 8(1):43–71, mar 1982.

[16] R.-E. Plessix, G. Chavent, and Y.-H. De Roeck. A quantitative kirchhoff migration to esimate the 2d velocity distribution. In *3rd Int. Conf. on mathematical and numerical aspects of wave propagation*, pages 704–712, Philadelphia, 1995. SIAM.

[17] R.-E. Plessix, Y.-H. De Roeck, and G. Chavent. Waveform inversion of reflection seismic data for kinematic parameters by local optimization. *SIAM journal of Scientific Computing*, 20(3):1033–1052, 1999.

[18] G. Quintana-Ortí, X. Sun, and C. H. Bischof. A blas-3 version of the qr factorisation with column pivoting. *SIAM J. Sci. Comput.*, 19(5):1486–1494, September 1998.

[19] Y. Saad. SPARSKIT: A basic tool kit for sparse matrix computations. Technical Report 90-20, Army High Performance Computing Research Center, Institute for Advanced Computer Science, NASA Ames Research Center, Moffett Field, CA, 1990.

[20] G.W. Stewart. Four algorithms for the efficient computation of truncated pivoted qr approximations to a sparse matrix. *Nümerische Mathematik*, 1999.

[21] W.W. Symes. The reflection inversion problem for acoustic waves. In *Mathematical and Numerical Aspect of the Wave Propagation Phenomena*, pages 423–433. SIAM, 1991.

[22] W.W. Symes and J.J. Carazzone. Velocity inversion by differential semblance optimization. *Geophysics*, 56(5):654–663, 1991.

[23] A. Tarantola. Linearized inversion of seismic reflection data. *Geophysical prospecting*, 32:998–1015, 1984.

[24] Ö. Yilmaz. *Seismic data processing*. Number 2 in Investigations in geophysics. Society of Exploration Geophysicists, Tulsa, 1987.

Unité de recherche INRIA Lorraine, Technopôle de Nancy-Brabois, Campus scientifique,
615 rue du Jardin Botanique, BP 101, 54600 VILLERS LÈS NANCY
Unité de recherche INRIA Rennes, Irisa, Campus universitaire de Beaulieu, 35042 RENNES Cedex
Unité de recherche INRIA Rhône-Alpes, 655, avenue de l'Europe, 38330 MONTBONNOT ST MARTIN
Unité de recherche INRIA Rocquencourt, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex
Unité de recherche INRIA Sophia-Antipolis, 2004 route des Lucioles, BP 93, 06902 SOPHIA-ANTIPOLIS Cedex

Éditeur
INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78153 LE CHESNAY Cedex (France)
http://www.inria.fr
ISSN 0249-6399